



90/005,628

REQUEST FOR REEXAMINATION

This is a request for reexamination of claims 1-15 of U.S. Patent 5,806,063.

New Issue of Patentability

The requester submits that a new issue of patentability is raised by any one of:

1. Ohms, "Computer Processing of Dates Outside the 20th Century", IBM Systems Journal, Volume 25, Number 2, 1986, pages 244-251;
2. Browe, "Intelligent Report Maintenance Using Dialogue Management", FOCUS Systems Journal, March 1990, pages 70-78;
3. Japanese Published Application 05-027947, February 5, 1993 and translation; and
4. The Millennium Journal, Volume II, Number IV, July 1995, pages 2-4;
5. Lysgaard, "The Time Bomb", IFIP TC8 Conference on Governmental and Municipal Information Systems, p. 513-519, 1987;
6. Shaw, "CAP Gemnni Tackles the Year 2000", NEWS 3X/400, June 1995, p. 30;
7. IBM, SAA AD/Cycle Language Environment, Programming Guide, Version 1, Release 3, March 1994; and
8. "SAS Language: Reference, Version 6. First Edition 1990.

Each of the foregoing references disclose the subject matter of at least one claim of the patent, none of these references were considered during the prosecution and each was published more than one year prior to the filing date of the application. Inasmuch as the only prior art rejection was withdrawn during prosecution, any pertinent reference would raise a new issue of patentability.

In the following portions of this request, the above identified references will be relied on, along with Shaughnessy U.S. Patent 5,630,118. Copies of these references are bound and filed herewith.

The Dickens Patent

The patent was issued on September 8, 1998 based on an application filed on October 3, 1996. The application succinctly (in five typed pages) purports to describe a solution to the year 2000 (or Y2K) problem. The problem arises from the conjunction of the use of two digits to designate a year, and the transition from the 20th to the 21st century. Even more plainly, consider whether the year "01" is before or after the year "98". By using only two digits to identify the year, it is impossible to tell whether 01 refers to 2001 or 1901.

The specification proposes a preferred date format of YYMMDD (where Y represents a year digit, M represents a month digit and D represents a day digit). Given a database with this format, the patent describes a solution as follow:

“A 10-decade window with a $Y_A Y_B$ value for the first year of the 10-decade window is selected, $Y_A Y_B$ being no later than the earliest $Y_1 Y_2$ year designator in the database. A century designator $C_1 C_2$ is determined for each date in the database, $C_1 C_2$ having a first value if $Y_1 Y_2$ is less than $Y_A Y_B$ and having a second value if $Y_1 Y_2$ is equal to or greater than $Y_A Y_B$. Each date in the database is formatted with the values $C_1 C_2$, $Y_1 Y_2$, $M_1 M_2$, and $D_1 D_2$.”

In other words, given a database or a collection of data which is restricted to a date range of no more than 10 decades, any 2-digit year date representation can be precisely located relative to the century boundary by selecting a pivot year (sometimes also called the base year) which is no larger than the earliest date represented in the database. This selection requires knowledge of the actual years represented in the database – i.e., the selection cannot be made based solely on the last two digits. Having made the selection, to locate any year we merely compare the two digit representation of the year in question ($Y_1 Y_2$) to the two digit representation of the pivot year ($Y_A Y_B$). If the year is smaller than the pivot year, then the date must be beyond the century boundary, i.e., for Y2K, the year is in the 21st century. This follows because we chose the pivot year to be as early as the earliest year in the collection. Once we determine that the year in question ($Y_1 Y_2$) is smaller than the pivot year, then by definition the year must be beyond the century boundary. Conversely, if the year is greater than or equal to the pivot, then the year is determined to be in the earlier century, i.e., the 20th century. The result again follow from the manner in which we chose the pivot.

For example, assume the database has data corresponding to dates from 1967 on. We select a pivot of 66 (we could have selected any number less than 66 or even 67, but no more than 67). Assume we retrieve 85 as the year in question. Since 85 is larger than 66 we determine that 85 refers to 1985 and not to 2085. Now assume we retrieve 42. We determine that 42 is less than 66 and so determine that 42 refers to 2042 and not to 1942.

This is windowing. It is claimed by Dickens. Significantly, however, it is also described by the prior art as will be described below.

The Prior Art in Context

Windowing, as just described, is equally well described by any of the eight references identified above, one of which (Ohms) anticipates the Dickens application by 8 years. Beyond windowing per se, as just described, the claims include additional subject matter apparently selected to give the appearance of claims with varying scope. In fact, as applied to Y2K the claims contain duplicative subject matter (such as claims 2-3 which specify that the dates being processed include the year 2000 and that the two centuries involved are the 20th and the 21st). Other claims contain peripheral subject matter (such as adding a sorting step subsequent to date processing, as if the connection between correct sorting and date processing was a contribution of the purported inventor) or, like claims 7 and 12, specify that date data can be converted from one format to another. And finally still other claims (8 and 13) pick out a particular set of years, apparently only because each member of the set is divisible by 10.

Windowing was so common that by the October 1994 publication date of Milam, "An Extended Date Library for C", C/C++ Users Journal, V 12, #10, pp 67-80, it was a standard function in a general purpose date processing library for application to instances of two digit year dates. See the portion of listing 6 appearing at p. 78. The general listing assumes "80" as the pivot and the comments in the listing includes the following note "anything less than 80 is considered to be in 21st century".

Even prior to Milam's 1994 publication however, the art was well acquainted with "windowing". In 1987 Kund Lysgaard presented "The Time Bomb" at the Conference on Governmental and Municipal Information Systems, in Budapest. Lysgaard's "Time Bomb" would later be referred to as the Y2K bug. Lysgaard recognized that given a date range of 100 years or less, the two digit representation of a year is theoretically adequate (p. 515). Lysgaard also recognized that all that was needed was the identification of the start year for the 100 year interval (Dickens refers to this parameter as Y_AY_B). Lysgaard continues:

"Information that the relevant interval starts in 1955 will for example, mean that 55 - 99 is interpreted as 1955 - 1999, whilst 00 - 54 is interpreted as 2000 - 2054." (p. 515)

As "windowing" became more widely known the descriptions became shorter and shorter. By June of 1995, Shaw's description was only a single sentence. He said (of Y2K solutions):

Another common solution is to pick a cut-off point, say 1950, where any two-digit dates after that point (51, 52 and so on) are treated as 20th century dates and any dates before that (01, 02, and so on) are considered post-millennium dates." "CAP Gemni Tackles the Year 2000", NEWS 3X/400, June 1995, p. 30

Of course, Shaw's "cut-off" is just Dickens' Y_AY_B. (Milam is bound with the other prior art).

Prosecution History

There are only two significant events in the prosecution history. On November 17, 1997 the Examiner issued an Office Action rejecting all fifteen claims on two different grounds. All claims were rejected as being anticipated by an IBM publication and all claims were rejected under 35 U.S.C. 112, first paragraph.

The response made no attempt to distinguish the claims from the IBM publication, rather that applicant purported to antedate the publication, alleging a reduction to practice in April 1996 (based on exhibit G) and alleging a conception prior to some unidentified date in or before October 1995.

Subsequent to filing the response, the Examiner withdrew the rejections and indicated that claims 1-15 were allowed. The Examiner noted:

“The prior art of record, taking into account the affidavit of the inventor, received 3/24/98, swearing behind the reference of the previous action, does not anticipate nor suggest the set of limitations of the claims, comprising the threshold year digits as used to determine a pair of century digits to be used for computation, but without enlarging the number of date digits of the database.” (Emphasis added).

Note there is nothing in the independent claims related to the emphasized portion of the Examiner’s statement. Furthermore, dependent claims 9 and 14 appear to contradict the Examiner’s statement.

The patent issued on September 8, 1998.

The Claims

The patent has 15 method claims, claims 1 and 11 are independent.

Claims 1 and 11 are generally similar, each including the step of:

providing a database which includes data using YY, MM and DD digits where “all of the symbolic representations of dates falling within a 10-decade period of time”.

Both claims also have a step of selecting a 10-decade window with a value ($Y_A Y_B$) for the first decade being no later than the earliest year in the database.

As disclosed in the specification, the window “may be arbitrarily selected” (col. 3, line 8). The first year of the selected window is $Y_A Y_B$ (col. 3, lines 13-14). “This selection process is performed in a completely automated fashion by the computer, without human input other than to select the starting date of the 10-decade window.” (col. 3, line 35-38; emphasis added). The “starting date” is of course the year which is also identified as $Y_A Y_B$. The human selection of $Y_A Y_B$ is also seen in Exhibit A (referred to at col. 3, line 58) inasmuch the parameter which is $Y_A Y_B$ in Exhibit A only appears “hard coded” in the line reading:

if $cl\$[1:2] < 50$ then

This if statement operates on the first two digits of the string $cl\$$, that is $cl\$[1:2]$ and determines if these digits are less than ($<$) the quantity 50. Thus $Y_A Y_B$ (or 50) has been selected by the person who wrote the code – and not the computer.

Since this disclosure (col. 3 lines 35-38 and Exhibit A) is the ONLY disclosure of the manner of selection of the window and/or $Y_A Y_B$ it follows that the “selection” clause must be interpreted to at least include human selection of the window and/or $Y_A Y_B$. Because of the paucity of disclosure supporting the “selection” clause it is correctly limited to human input.

Both claims have a step of determining a century designator by comparing a year designator ($Y_1 Y_2$) with $Y_A Y_B$ and then reformatting the symbolic representation of the dates using the determined value of $C_1 C_2$. The specification describes one way to “reformat” the

symbolic representation (at col. 3, lines 41-43) which is to actually prepend C_1C_2 to $YYMMDD$. This reformatting expressly described in claim 11. Claim 1 appears to be broader since it only requires the use of C_1C_2 in the reformatting. In other words while claim 11 specifies the format produced by the reformat operation, claim 1 only requires the parameter C_1C_2 be used in the reformat operation. However, since the specification fails to describe any other way to use C_1C_2 in reformatting the symbolic representation of the date (as is claimed in claim 1) it is not apparent there is any difference in claims 1 and 11 in this regard. In other words, while on its face claim 1 appears to have a broader "reformat" clause than claim 11, in fact the paucity of disclosure requires both clauses to have identical scope.

Claim 11 also includes an additional step of "sorting the dates" using the recited format.

Claims 1 and 11 require interpretation because each includes a step of "selecting a 10-decade window with a $Y_A Y_B$ value for the first decade of the window". Considering only the claim language, it is not apparent what the "value" for a decade might be. A decade by definition includes 10 years, each of which has, or represents, a different value.

Dependent claim 8 specifies that $Y_A Y_B$ is selected such that Y_B is zero. This suggests that whatever the "value" of the decade recited in claim 1 is, it is not necessarily a 2-digit number divisible by 10. Reference to the specification fails to uncover any definition for the phrase "selecting a ... window with a ... value". The specification describes a 10-decade window where $Y_A Y_B$ is the "value for the first year" of the window. If claims 1 and 11 called for selecting such a value, there would be no need for interpretation. The best that can be said is that the claims should be interpreted to mean that $Y_A Y_B$ is the value for the first year of the 10-decade period of the window. The claims will be so interpreted in the following portions of this document.

Claims 1 specifies"

"determining a century designator ... for each symbolic representation of a date in the data base ..." and "reformatting the symbolic representation of the date".

Claim 11, in a similar fashion specifies:

"determining a century designator ... for each date in the data base" and "reformatting each date ...".

While these claims appear to require the determining and reformatting steps to be performed for "each" element of the data base, they do not specify either (1) the time period over which the steps are applied or (2) the order in which these steps are to occur. In other words, the determining and reformatting steps might be applied to 25 % of the data base at one time, and to the remaining 75% of the data base at another time, or to disparate parts of the data base at different times. In addition, the determining step might be applied to a set A of the data, and the following reformatting step could be applied to only a part of set A of the data. At a later time the determining and reformatting steps are applied to all of the non-set A data. Finally at a still later time the reformatting step may be applied to that part of the set A data not yet reformatted. As will become clear each of the references describe procedures to eliminate data ambiguity as a consequence of the conjunction of the turn of the century and the use of two digit-year dates.

The procedures which are described in each of the references are taught to be applied to each element of the associated data base, just as in these claims. Consequently the claims do not distinguish from the references through the use of the term "each" in the claims.

Finally, claim 1 has to be interpreted to properly read the "providing" clause. That clause requires:

providing a database with symbolic representations of dates stored therein according to a format wherein M_1M_2 is the numerical month designator, D_1D_2 is the numerical day designator, and Y_1Y_2 is the numerical year designator, all of the symbolic representations of dates falling within a 10-decade period of time;

The claim expressly says that it is the symbolic representations which fall within a 10-decade window. Contrast claim 1 with claim 11 which notes that it is the dates which fall within the 10-decade window. It should be apparent (with respect to claim 1) that the two-digit year representations, Y_1Y_2 (only two digits) by definition cannot fall outside a 10-decade period of time.

References

Collectively the references span almost a decade, e.g., beginning with the 1986 Ohms publication and ending with the July 1995 Millennium Journal. Each of the references is concerned with date processing in the context of Y2K.

Ohms

The 1986 Ohms publication is expressly directed at date processing occasioned by the end of the 20th century, e.g., the Y2K problem. Ohms in general describes conversions between different date formats. The fact that he contemplates applying these conversions to databases using YYMMDD data to express dates is readily identified by the table on page 247. The table indicates that Ohms describes a function to convert a "short Gregorian" date (having the format YYMMDD). Ohms succinctly describes the subject matter claimed by Dickens in the first full paragraph of the right hand column on page 248. Ohms states:

"However, it may be necessary to provide a conversion function that receives a definition of the implied century as a parameter. An excellent way to do this unambiguously is to specify a year as the desired starting point of a 100-year range. For example, if the starting year for the range is specified as 1925, dates with year digits of 25 through 99 would be between 1925 and 1999, and dates with year digits of 00 through 24 would lie between 2000 and 2024."

Ohms emphasizes the caution against using this procedure where the date range spans more than 100 years by indicating (at page 249 in the left hand column) "Where systems contain dates that span a range of more than 100 years, the century must have already have been carried. In the rare event that this is not true, immediate conversion is unavoidable". Ohms like Dickens

works with a 10-decade window (10-decades is identically 100 years). It should be emphasized that the dates within Ohms' 100 year range typically fall into two different centuries (just as in Dickens). The Ohms "starting point" (which is the same as the $Y_A Y_B$ parameter of Dickens) defines a 10-decade window (or 100 year range). It should be noted that even in lines 28-38 on p. 248 Ohms is still referring to the 100 year range (which is identically the Dickens 10-decade window). Ohms, in the same passage refers to a "beginning date" (which is the same as the "starting point") "set eighty years prior to the current systems date". This sentence relates the "beginning date" or "starting point" to a current date (the current systems date). This duration (eighty years in the text) has *nothing* to do with the duration or extent of 10-decade window (or 100 year range). In this passage Ohms relates the 100 year window to the present, so as to indicate how far into the future the window extends. The Ohms example in which the start year is eighty years in the past also means that the data base accommodates data for 20 years into the future (since the total span is 100 years). We will see other references also working with this relationship. All of the references which work with this relationship are more comprehensive than Dickens because *Dickens never mentions this relationship*.

Lysgaard

Lysgaard describes, in "The Time Bomb" presented at the 1987 Conference on Governmental and Municipal Information Systems, in Budapest, his solution to the problem caused in EDP by the use of two digit dates as the year 2000 approaches. Lysgaard's "Time Bomb" would later be referred to as the Y2K bug. Lysgaard stated "If at all times a date has a relevant range of less than (or equal to) 100 years, then a two digit year is theoretically adequate to identify the year within the relevant interval of time" (p. 515). Lysgaard also recognized:

"If information as the valid time interval is added to the programme - maybe just the start year for the 100 year interval - the programmes will be able to handle all time calculations correctly."

What Lysgaard called the "start year" is the same parameter that Dickens refers to as $Y_A Y_B$. This follows since there is no data prior to the "start year" in the data base. This also meets Dickens' prescription that no year representation be earlier than $Y_A Y_B$. Lysgaard continues:

"Information that the relevant interval starts in 1955 will for example, mean that 55 - 99 is interpreted as 1955 - 1999, whilst 00 - 54 is interpreted as 2000 - 2054." (p. 515)

Lysgaard also calls attention to the errors which result when sorting using two digit years (p. 516). One solution he proposes is the "temporary addition of auxiliary fields stating the century and included in the sorting criteria" (p. 516). This is exactly the Dickens solution. Dickens prefers to call the "temporary addition of auxiliary fields" a "reformatting", but however you name the operation it is the prepending of YYMMDD with CC, referencing the correct century for the date and then a straight numerical sort on the augmented (CCYYMMDD) data.

Browe

Browe describes a date utility, which perform several functions in the context of Y2K. At the bottom of page 70 a comment to the Browe software describes the purposes of REFDATE and indicates that "the reference date is typed in MMDDYY format and must refer to a date from 01/01/1950 to 12/31/2049 for this FOCEXEC to work correctly." In other words, the date range must be no more than 10-decades. On the next page (page 71) Browe describes how 2-digit years are properly interpreted and reformatted to take into account the transition from the 20th to the 21st century. The logic is shown in detail as follows:

"IF &YR GE 50 AND &YR LE99 THEN 19/&YR
ELSE 20/&YR".

This can be expressed in text as follows: if the 2-digit parameter &YR is greater than or equal to (this is the meaning of GE) 50, and the same parameter is less than or equal to (this is the meaning of LE) 99, then change the format of the year from the two digit parameter &YR to the four digits 19&YR, otherwise reformat the 2-digit year date to the four digits 20&YR.

As an example, if the parameter &YR is 42 (not between 50 and 99), then this logic will reformat the year as 2042 and if the &YR parameter was 67 (between 50 and 99), then this logic would format the year as 1967.

Note that Browe uses 50 as the test parameter (Y_AY_B in terms of Dickens). This is the earliest date in the range so that there are no earlier dates.

Published Japanese Patent Publication 05/027,947

Attached to this request is both the Japanese patent publication itself, along with an accurate translation thereof.

The Japanese patent publication is also directed at Y2K. The translation (page 2) notes, the purpose is:

"To guarantee the year order, even for years after 2000AD, with the current file format, even when the year is managed by the last 2 digits of the date in digital files."

The Y2K problem is described at page 3 of the translation where the text indicates that systems using the last two digits of the years to indicate dates do not take into consideration years after 2000AD. The text continues that:

"When ascending/descending order is handled by processing that evaluates magnitude and by sort/merge processing using normally numbered years, their relative magnitudes are represented by formula 1

1999 > 1998 > 2001 > 2000."

In other words prior art systems focusing only on years expressed as two digits will, when sorting the four years 1998-2001, produce a result which indicates that the latest year is 1999, which is preceded by 1998 which is preceded by 2001 which is preceded in turn by 2000. This clearly is incorrect. On page 4, the text indicates that the invention provides a method of guaranteeing the proper year order and indicates that in general when there are data present that indicate years in the 1900s and 2000s AD, the data code that represents the date is replaced by another code so the year order will be maintained. The manner in which this is accomplished is described at the bottom of page 4. A module 10 is activated to effect preprocessing in order to handle the year calculations. The text indicates:

“Note that, in module (10), a range of the last 2 digits for which code transformation will be performed are specified in advance. Replacement involves numbers for years in the 2000’s, where the last two digits are smaller than the smallest number in the last 2 digits in the years in the 1900’s. For example, when data in file (6) for years AD begin with the year 1973, the last two digits are replaced using 00 (year 2000) for 72 (year 2072). The present application example is an example where there are data from year 1960 in file (6), such a range is specified so that the last 2 digits will be transformed to codes 00-59”.

In other words, year data which is “smaller than the smallest number in the last two digits in the years in the 1900s” are recoded to indicate dates in the 21st century. Of course Dickens called recoding, reformatting. What the translation refers to as “the smallest number in the last two digits in the years in the 1900’s”, Dickens labeled $Y_A Y_B$. The two examples employed illustrate this processing is necessarily limited to 10-decade ranges. The text indicates when the input file has year dates beginning with the year 1973 ($Y_A Y_B=73$), the data is recoded to indicate dates up to and including 72 lie in the 21st century. Similarly, when the lowest dates in the input file is 1960 ($Y_A Y_B=60$) dates 00-59 are recoded (or reformatted) indicate they lie in the 21st century.

The Millennium Journal

From its title it is clear that this reference also deals with Y2K. The extract that is relied on here describes a variety of date formats that are employed. Most pertinent is the text under the heading “logic-based century determination”. The text indicates that this is also referred to as windowing. The text indicates:

“In windowing, the two digit years are left alone in the files. A base year is selected (e.g., “50”) where every year starting with (or, in some cases, greater than) “50” ($Y_A Y_B=50$) through “99” is treated as a 1900 date, and any year less than (or equal to) “50” is treated as 2000.”

The very next paragraph indicates that windowing is associated with sorting. The text indicates that when a subroutine is used to make the specified format change, an exit must be made before sorting is implemented.

SAA AD/Cycle Language Environment

The IBM, SAA AD/Cycle Language Environment, Programming Guide, Version 1, Release 3, March 1994, describes the 370 Language Environment including callable services. Date and Time callable services are summarized at p. xvi. In addition to various date format conversions, two of those services provide for queries to determine the century within which the Language environment assumes that two digit dates lie (CEEQN – p. 146) and setting the century (CEESSEN – p. 149). The manner in which format conversion handles ambiguous two digit years is explained at p. 84.

By default, 2-digit years lie within the 100-year range starting 80 years prior to the system date. Thus, in 1993, all 2-digit years represent dates between 1913 and 2012, inclusive. This default range is changed by using the callable service 'CEESCEN-Set the Century Window' on page 149".

A little thought will reveal that if all 2 digit dates fall between 1913 and 2012, then the system uses 13 as the "start year" (following Lysgaard or Ohms) and assigns "19" as C1 if the year is greater than or equal to 13 and assigns "20" as C2 if the year is less than "13". If there was any doubt that this environment is related to the YYMMDD format, that doubt is dissipated by the disclosure at p. 149, under the heading CEESCEN

This is another example in which the text relates the 100 year window to the present. The default range (p. 84) is identified as beginning 80 years prior to the present. As noted with respect to Ohms, Dickens does not mention the relationship of the 100 year window to the present. What is important is that, to the extent there is a description of a 10 decade window in Dickens, *that description is anticipated by this reference*, as well as the other references described here.

SAS Language: Reference

"SAS Language: Reference", Version 6. First Edition 1990 describes facilities of the SAS system. The informats listed on p. 63 have several date variations such as ddmmyy, ddmmmy, mmdyy, mmyy and yymmdd. The yy, mm and dd digits used in this reference have the identical meanings to those used by Dickens. The mmm digits refer to the use of three alpha characters to identify a month, also as in Dickens. At p. 129, the manual describes the protocol used with 2-digit year values. It says:

Two-digit years are attributed to the century specified by the YEARCUTOFF= system option. The YEARCUTOFF= system default is 1900, allowing the years 1900 through 1999 to be specified as two-digit year values. You can also override the system default and specify a beginning date of your choice.

Of course the "beginning date" is the same as Ohms' "starting point", Lysgaard's "start year" and Dickens' Y_AY_B. SAS Language: Reference also refers to sorting (p. 131) emphasizing the importance to sorting of correct date interpretation, such as by the use of the YEARCUTOFF parameter. The YEARCUTOFF parameter, its use and associated facilities is described in detail

at p. 790. Instead of using the phrase "beginning date", the description at p. 790 refers to the "first year of the 100 year span". This is clearly the same as Dickens' "value for the first year of the 10-decade period of time" which is the definition of $Y_A Y_B$ found at col. 1, lines 63-65 of the Dickens patent. Fig. 16.1 (p. 791) which illustrates the 100 year span and the YEARCUTOFF parameter could apply to Dickens as well as to the SAS system.

Shaw

Shaw, in "CAP Gemnni Tackles the Year 2000", NEWS 3X/400, June 1995, p. 30, describes Y2K "windowing" in a single sentence. He said:

Another common solution is to pick a cut-off point, say 1950, where any two-digit dates after that point (51, 52 and so on) are treated as 20th century dates and any dates before that (01, 02, and so on) are considered post-millennium dates.

Of course, the "cut-off" is the same parameter that Dickens labels as $Y_A Y_B$. "Considering" the two digit dates as either 20th or 21st century dependent on the relation between the "cut-off" and the two digit year is the same as the "reformatting". Recall that YEARCUTOFF is also the parameter used in SAS Language: Reference, Version 6.

Shaughnessy

The Shaughnessy patent relates to computer systems that perform date operations on date fields spanning a century boundary. While the title is generic, the text indicates that it is Y2K which is the genesis of the patent. Shaughnessy describes the modifications to computer systems so that date operations can be performed correctly even when processing dates after December 31, 1999, e.g., Y2K. The data formats that are employed in accordance with Shaughnessy are found in a table attached as an appendix, see column 18. As indicated in column 18, there are several different formats that are represented as "YYMMDD". The appendix notes that for formats B, F and S (all of which are YYMMDD) that "the date cycle is 100 years". The general sequence of operations described by Shaughnessy is shown in figure 2 where the requested date operation is performed only after certain precursor steps are performed. The precursor steps include determining the current date, determining the end of the 100-year cycle and determining two possible century values. As shown in figure 4, the end of the 100-year cycle can be determined either based on the current date or based on the system installation date. In either event, once the end of the 100-year cycle is determined, the system derives two dates separated by a period of 100 years (10 decades).

Figure 5 shows how the two possible century values are determined. In particular, the later of the two centuries is determined as the century of the date at the end of the 100-year cycle, and the earlier century is the century preceding the later century.

Figure 7 shows how the century value (this corresponds to $C_1 C_2$ of Dickens) is assigned. Assignment includes a comparison between the date representation (e.g., YY) with a "end of the 100 year cycle". The text indicates that:

"If the date is less than or equal to the end of the 100 year cycle date, the CENTURY2 value is assigned to the date (box 64). If the date is greater than the end of cycle date, the CENTURY1 value is assigned to the date (box 66)". Column 7, lines 9-13.

Note this Shaughnessy century determination is identical to the Dickens century determination. This identity is apparent by equating Dickens' $Y_A Y_B$ with the Shaughnessy "end of the 100 year cycle date".

After the century designator (this is CENTURY1 or CENTURY2) is assigned the date is reformatted to the format YYYYMMDD (see column 6 line 65). Of course, the first two of the Y digits represent the assigned century. This is identical to the format $(C_1 C_2 Y_1 Y_2 M_1 M_2 D_1 D_2)$ of Dickens claim 11.

While the Shaughnessy specification comprehends several embodiments, the embodiment which deals with a C1 date format is limited to date spans of 100 years or less. This is apparent from the determination of the "end of the 100 year cycle" and the fact the date ambiguity is resolved by determining which of two possible century values is appropriate.

Shaughnessy indicates (column 1, line 26 and column 8, line 34 – column 12, line 20) that computers typically compare dates. Indeed, sequentially comparing dates is also referred to as sorting and therefore in these passages Shaughnessy teaches that dates processed in accordance with the procedure just described can then be used for sorting purposes.

While the impetus for the Shaughnessy patent was Y2K (the problem occasioned by the transition from the 20th to the 21st century) it should be clear that the Shaughnessy specification also describes solutions applicable to analogous but not necessarily identical problems. The Shaughnessy specification describes (1) maintaining a database unchanged (1/60-2/5) even though the data may lead to ambiguity and (2) the alteration of the program logic by the addition of a subroutine. The subroutine allows the selection of the one appropriate date when the stored data is otherwise ambiguous. Reference to the Appendix (col. 18-19) indicates that Shaughnessy contemplated one data format in which the date "cycle" was 100 years (the C1 format) and another data format in which the date "cycle" was 10000 years (C2). Shaughnessy describes that using a two digit year data format, 100 year cycle data becomes ambiguous at the turn of the century (that is the reason it is necessary to determine whether a date is CENTURY1 or CENTURY2). Using four digit year data there is a similar ambiguity in 10000 year cycle data, i.e., when there is a transition from the year 9999 to the year 10000. While the Shaughnessy specification may apply to databases with C1 data, as well as database with C2 data, there is nothing in the patent to suggest mixing 100 year cycle data with 10000 year cycle data in one database. It should be clear from the description of "windowing" contained in Ohms (1986), Lysgaard (1987), Browe (1990), SAS Language Reference Manual (1990), SAA AD/Cycle Language Environment (1994) and the Japanese Patent Publication 05-027947.(1993) that by the 1994 filing of the Shaughnessy application the art was well aware of the 100 year limit associated with two digit year data. Consequently the Shaughnessy description is quite adequate on the question of limiting data ranges to 100 years when resolving date ambiguity with two digit year data. The first Shaughnessy flowchart has a function to determine the "end of 100-year cycle" – this discloses the 100 year limit of year data to those skilled in the art as of 1994.

Shaughnessy, like some of the other references also relates the 100 year window to the present. This relationship is determined by the "number of years of future dating" (6/10). Shaughnessy also provides another degree of freedom to the user in that the "end of the 100 year cycle" may be updated (6/13). In both respects Shaughnessy goes beyond the Dickens specification in that *Dickens never mentions either parameter*. Neither of these parameters ("number of years of future dating" or the ability to update the "end of the 100 year cycle") bears on the manner in which Shaughnessy anticipates the Dickens 100 year window or the manner in which the correct century designator is determined. To be sure Shaughnessy describes using the "end of the 100 year cycle" while Dickens uses the beginning of the 100 year cycle. However, for any given window these two numbers differ by unity. Consider a Dickens window defined by $Y_A Y_B$ of 50. This window extends from 1950 to 2049. The corresponding Shaughnessy "end of the 100 year cycle" is 49, and it defines the same window, from 1950 to 2049.

Application of the References

Attachments 1-7 are a series of claim charts applying the terms of the claims to the references. As evidenced by the claim charts the Requestor contends that:

Ohms anticipates claims 1-3, 7, 9 and 10, [Attachment 1],

Shaughnessy anticipates claims 1-7, 11 and 12 [Attachment 2],

Lysgaard anticipates claims 1-15 [Attachment 3], and

Browe anticipates claims 1-3, 5 and 8 [Attachment 4].

The Requestor further contends that:

Ohms considered with the Japanese Published application ('947) invalidates (under 35 USC 103) claims 4, 11 and 13-15 [Attachment 5],

Shaughnessy considered with the Japanese Published application ('947) invalidates (under 35 USC 103) claims 4, 6, 8 and 13 [Attachment 6], and

Ohms considered with the Millennium Publication invalidates (under 35 U.S.C. 103) claims 5, and 11 [Attachment 7].

Shaughnessy taken with '947

Claims 8 and 13 have in common a step of selecting $Y_A Y_B$ such that Y_B is zero (0), in other words $Y_A Y_B$ is divisible by ten. Whether or not this specificity amounts to an unobvious difference (over selecting any other value for $Y_A Y_B$) is academic in this context inasmuch as one of the examples in '947 is the use of the quantity "60" for the parameter which corresponds to $Y_A Y_B$. Claims 1 and 11 (the parent claims for claims 8 and 13, respectively) are each anticipated by the Y2K related disclosure of the Shaughnessy patent. '947 discloses another

Y2K solution. The disclosure of '947 shows (if any showing is necessary) an example of the use of a parameter corresponding to $Y_A Y_B$ which is divisible by ten. Because of the relation between the subject matter of Shaughnessy and '947 (both Y2K related) those skilled in the art would naturally view the two together and as such note the '947 suggestion of using a parameter corresponding to $Y_A Y_B$ which is divisible by ten as applicable to Shaughnessy. Requestor submits that the foregoing demonstrates that it would have been obvious in the sense of 35 USC 103 at the time ('947 was published in 1993 and Shaughnessy has a 1994 filing date) to use a parameter corresponding to $Y_A Y_B$ which is divisible by ten in the method described by Shaughnessy.

Claims 4, 6 have in common a step of "sorting" subsequent to the "reformatting" step of the parent claim. '947 discloses the benefit to sorting operations from Y2K remediation. The parent claims 1 and 5, respectively are anticipated by Shaughnessy. Both Shaughnessy and '947 relate to Y2K remediation or correction. Because of the relation between the subject matter of Shaughnessy and '947 (both Y2K related) those skilled in the art would naturally view the two together and as such note the '947 teaching of the benefits to sorting operations from Y2K remediation as applicable to the Shaughnessy Y2K remediation. Thus to the extent the disclosure of Shaughnessy fails to anticipate claims 4 and 6, consideration of '947 overcomes the validity of these claims. Requestor submits that, for the reasons outlined above, claims 4 and 6 describe subject matter which would have been obvious to those skilled in the art at the time ('947 was published in 1993 and Shaughnessy has a 1994 filing date).

Ohms taken with '947

Claims 4 and 11, as well as 13, 14 and 15 (dependent on claim 11) contain a step of "sorting" subsequent to a "reformatting" step. This subject matter is absent from claim 1. '947 discloses the benefit to sorting operations from Y2K remediation. Claim 1 is anticipated by Ohms. In addition claims 11 and 13-15 contain a more specific "reformatting" step than that found in claim 1. As evidenced by the attachment 5 however, this more specific "reformatting" subject matter is also anticipated by Ohms. Both Ohms and '947 relate to Y2K remediation or correction. Because of the relation between the subject matter of Ohms and '947 (both Y2K related) those skilled in the art would naturally view the two together. Consequently those of skill would note the '947 teaching of the benefits to sorting operations from Y2K remediation and understand those same benefits would flow from the Ohms Y2K remediation. Thus to the extent the disclosure of Ohms fails to anticipate the "sorting" subject matter of claims 4, 11 and 13-15, consideration of '947 overcomes the validity of these claims with respect to the "sorting" subject matter. Requestor submits that, for the reasons outlined above, claims 4, 11 and 13-15 describe subject matter which would have been obvious to those skilled in the art at the time ('947 was published in 1993 and Ohms was published in 1986).

Ohms taken with The Millennium Journal

Claims 5 and 11 differ from claim 1 by specifying the format of the reformatted representation as opposed to claim 1 which only recounts the data involved in the reformat operation. Both Ohms and the Millennium Journal deal with Y2K remediation. Because of the relation between their disclosures those skilled in the art at the time (Ohms has a 1986

publication date and The Millennium Journal has a July 1995 publication date) would have naturally viewed the two disclosures together. The Millennium Journal specifies a number of data formats used in connection with Y2K remediation, one of those formats is the specific format called for in claims 5 and 11. To the extent, if any, by which Ohms fails to anticipate claims 5 and 11, consideration of the Millennium Journal, and specifically the disclosure therein of the specific format recited in claims 5 and 11, reveals that those skilled in the art at the time would have taken from The Millennium Journal knowledge that use of the format claimed in claims 5 and 11 was extant. That knowledge, coupled with the other teachings of Ohms would have rendered the subject matter of claims 5 and 11 obvious (35 U.S.C. 103) to those skilled in the art at the time, thus rendering it evident that claims 5 and 11 are invalid.

Conclusion

For the reasons expressed, Requestor submits that each and every claim of patent 5,806,063 is invalid as anticipated or is such as would have been obvious to those skilled in the art at the appropriate time.

00005628-020200

<p>Claims of 5806063</p> <p>1. A method of processing symbolic representations of dates stored in a database, comprising the steps of</p>	<p>Ohms</p> <p>Ohms describes a "date processing method" (p. 244) as well as a "conversion function" (p. 248)</p>
<p>providing a database with symbolic representations of dates stored therein according to a format wherein M₁ M₂ is the numerical month designator, D₁ D₂ is the numerical day designator, and Y₁ Y₂ is the numerical year designator, all of the symbolic representations of dates falling within a 10-decade period of time;</p>	<p>see the table on p 247 including reference to "a short Gregorian date", the conversion function works with a number of data formats including this one, Ohms describes, at p. 249, the 100 year limitation (identical to 10-decades)</p>
<p>selecting a 10-decade window with a Y_A Y_B value for the first decade of the window, Y_A Y_B being no later than the earliest Y₁ Y₂ year designator in the database;</p>	<p>see p. 248, right hand column – specify a year as the desired starting point of the range – this is Y_A Y_B, and it is no later than any year in the data base</p>
<p>determining a century designator C₁ C₂ for each symbolic representation of a date in the database, C₁ C₂ having a first value if Y₁ Y₂ is less than Y_A Y_B and having a second value if Y₁ Y₂ is equal to or greater than Y_A Y_B, and</p>	<p>the century designation is determined by comparing the year date with the desired starting point, if the year in question is greater then the century is the earlier one and vice versa, see p. 248</p>
<p>reformatting the symbolic representation of the date with the values C₁ C₂, Y₁ Y₂, M₁ M₂, and D₁ D₂ to facilitate further processing of the dates.</p>	<p>the "implied century" (see p. 248, right hand column) which is the result of the date "conversion" is the reformatted date, note this claim does not require any specific format as long as the four parameters are used.</p>
<p>2. The method of claim 1, wherein the 10-decade window includes the decade beginning in the year 2000.</p>	<p>See p. 248, the dates determined are in the 20th and 21st centuries and so include the decade including the year 2000.</p>
<p>3. The method of claim 2, wherein the step of determining includes the step of</p>	
<p>determining the first value as 20 and the second value as 19.</p>	<p>See p. 248, the dates determined are in the 20th and 21st centuries</p>

Claims of 5806063	Ohms
7. The method of claim 1, wherein the step of providing a database includes the step of	
converting pre-existing date information having a different format into the format wherein M ₁ M ₂ is the numerical month designator, D ₁ D ₂ is the numerical day designator and Y ₁ Y ₂ is the numerical year designator.	Ohms describes a number of date format conversions as well as the use of the specific YYMMDD format, the table on p. 247 refers to one conversion to YYMMDD
9. The method of claim 1, including an additional step, after the step of reformatting, of	
storing the symbolic representation of dates and their associated information back into the database.	See p. 248-249 where Ohms discusses the storage issue and expressly mentions storing date data with four digit year representations.
10. The method of claim 9, including the additional step, after the step of reformatting, of	
manipulating information in the database having the reformatted date information therein.	Ohms refers to the need for date conversions for handling future applications (p. 250). Future applications for data base programs require manipulation of the data.

5005628-020200

'063 claims	Lysgaard
1. A method of processing symbolic representations of dates stored in a database, comprising the steps of	the YY year data (see below) is a symbolic representation and the interpretation (p. 515) of this symbolic data corresponds to the processing
providing a database with symbolic representations of dates stored therein according to a format wherein $M_1 M_2$ is the numerical month designator, $D_1 D_2$ is the numerical day designator, and $Y_1 Y_2$ is the numerical year designator, all of the symbolic representations of dates falling within a 10-decade period of time;	data bases store dates with 2 digits representing the year (p. 513); the paragraph bridging pp 517-18 makes it clear that the typical data base stores yy, mm and dd data for a date, p. 515 – "if at all times a date has a relevant range of less than (or equal to) 100 years" this is the direction to insure the database spans no more than 100 years
selecting a 10-decade window with a $Y_A Y_B$ value for the first decade of the window, $Y_A Y_B$ being no later than the earliest $Y_1 Y_2$ year designator in the database;	information as to the valid time interval, such as the start year for the 100 year interval (p. 515), the start year corresponds to $Y_A Y_B$, by definition there is no year data ($Y_1 Y_2$) in the database earlier than the "start year"
determining a century designator $C_1 C_2$ for each symbolic representation of a date in the database, $C_1 C_2$ having a first value if $Y_1 Y_2$ is less than $Y_A Y_B$ and having a second value if $Y_1 Y_2$ is equal to or greater than $Y_A Y_B$, and	the interpretation of the data (see p. 515), where 55-99 is interpreted as 1955-1999 while 00-54 is interpreted as 2000-2054 comprehends determining the century designator on the basis recited in the claim
reformatting the symbolic representation of the date with the values $C_1 C_2$, $Y_1 Y_2$, $M_1 M_2$, and $D_1 D_2$ to facilitate further processing of the dates.	the reformatting is the same as the "temporary addition of auxiliary fields stating the century" (p. 516) by prepending the century designator (19 or 20)
2. The method of claim 1, wherein the 10-decade window includes the decade beginning in the year 2000.	the abstract and initial heading refer to the year 2000, and the example on p. 515 includes the decade beginning in the year 2000
3. The method of claim 2, wherein the step of determining includes the step of determining the first value as 20 and the second value as 19.	see the example on p. 515, where the first value is associated with the year dates 00-54 and the second is associated with the dates 55-99
4. The method of claim 1, including an additional step, after the step of reformatting, of	
sorting the symbolic representations of dates.	the paragraph entitled "sorting" on p. 516 makes it clear that sorting can be accomplished after the proper date interpretation.

5. The method of claim 1, wherein the step of reformatting includes the step of reformatting each symbolic representation of a date into the format C ₁ C ₂ Y ₁ Y ₂ M ₁ M ₂ D ₁ D ₂ .	the reformatting is the same as the "temporary addition of auxiliary fields stating the century" (p. 516) by prepending the century designator (19 or 20)
6. The method of claim 5, including an additional step, after the step of reformatting, of sorting the symbolic representations of dates using a numerical-order sort	the paragraph on p. 516, entitled "sorting" describes the sorting operation after the "temporary addition", the fact that it is a numerical order sort is made clear by the description of the sort order (6789012345) for the period 1960-2059 at p. 516
7. The method of claim 1, wherein the step of providing a database includes the step of converting pre-existing date information having a different format into the format wherein M ₁ M ₂ is the numerical month designator, D ₁ D ₂ is the numerical day designator and Y ₁ Y ₂ is the numerical year designator.	the text indicates (p. 516) that most date manipulations take place with standard routines, at p. 518 the text notes that the standard routines also handle conversion between different formats
8. The method of claim 1, wherein the step of selecting includes the step of selecting Y _A Y _B such that Y _B is 0 (zero).	the paragraph on sorting (p. 516) describes an example where the "start year" is at the beginning of a decade (60) such that the parameter corresponding to Y _B is 0
9. The method of claim 1, including an additional step, after the step of reformatting, of storing the symbolic representation of dates and their associated information back into the database.	the step of storing is tantamount to converting a 2 digit year data base to one using 4 digit years, this is described at p. 518 where the text notes that "standard routines also include conversion between various date formats – for example dates which are exported from an old system (shortened year) to a new system (full year)".
10. The method of claim 9, including the additional step, after the step of reformatting, of manipulating information in the database having the reformatted date information therein.	the clause relates to any use (manipulating) of the data after a "reformat" has been stored, use of the data base is the purpose of any data base and so this step is inherent in any data base,

	such as that operated on by the Lysgaard method
11. A method of processing dates in a database, comprising the steps of	the YY year data (see below) is a date representation and the interpretation (p. 515) of this data corresponds to the processing
providing a database with symbolic representations of dates stored therein according to a format wherein M ₁ M ₂ is the numerical month designator, D ₁ D ₂ is the numerical day designator, and Y ₁ Y ₂ is the numerical year designator, all of dates falling within a 10-decade period of time which includes the decade beginning in the year 2000;	data bases store dates with 2 digits representing the year (p. 513); the paragraph bridging pp 517-18 makes it clear that the typical data base stores yy, mm and dd data for a date, p. 515 – “if at all times a date has a relevant range of less than (or equal to) 100 years” this is the direction to insure the database spans no more than 100 years, page 513 makes it clear that the problem is caused by the year 2000 and thus this year and the decade including it are included in the data
selecting a 10-decade window with a Y _A Y _B value for the first decade of the window, Y _A Y _B being no later than the earliest Y ₁ Y ₂ year designator in the database;	“information as to the valid time interval”, such as “the start year” for the 100 year interval (p. 515), the start year corresponds to Y _A Y _B , by definition there is no year data (Y ₁ Y ₂) in the database earlier than the “start year”
determining a century designator C ₁ C ₂ for each symbolic representation of a date in the database, C ₁ C ₂ having a first value if Y ₁ Y ₂ is less than Y _A Y _B and having a second value if Y ₁ Y ₂ is equal to or greater than Y _A Y _B ;	the interpretation of the data (see p. 515), where 55-99 is interpreted as 1955-1999 while 00-54 is interpreted as 2000-2054 comprehends determining the century designator on the basis recited in the claim
reformatting each date in the form C ₁ C ₂ Y ₁ Y ₂ M ₁ M ₂ D ₁ D ₂ to facilitate further processing of the dates; and	the reformatting is the same as the “temporary addition of auxiliary fields stating the century” (p. 516) by prepending the century designator (19 or 20)
sorting the dates in the form C ₁ C ₂ Y ₁ Y ₂ M ₁ M ₂ D ₁ D ₂ .	that this processing is the predicate for sorting is described at p. 516, under the heading “sorting”
12. The method of claim 11, wherein the step of providing a database includes the step of converting pre-existing date information having a different format into the format wherein M ₁ M ₂ is the numerical month designator, D ₁ D ₂ is the numerical day designator, and Y ₁ Y ₂ is the numerical year designator.	the text indicates (p. 516) that most date manipulations take place with standard routines, at p. 518 the text notes that the standard routines also handle conversion between different formats

13. The method of claim 11, wherein the step of selecting includes the step of selecting Y_A Y_B such that Y_B is 0 (zero).	the paragraph on sorting (p. 516) describes an example where the "start year" is at the beginning of a decade (60) such that the parameter corresponding to Y_B is 0
14. The method of claim 11, including an additional step, after the step of sorting, of storing the sorted dates and their associated information back into the database.	the storing step is tantamount to converting a 2 digit year data base to one using 4 digit years, this is described at p. 518 where the text notes that "standard routines also include conversion between various date formats – for example dates which are exported from an old system (shortened year) to a new system (full year)".
15. The method of claim 14, including the additional step, after the step of sorting, of manipulating information in the database having the reformatted date therein.	the clause relates to any use (manipulating) of the data after a "reformat" has been stored, use of the data base is the purpose of any data base and so this step is inherent in any data base, such as that operated on by the Lysgaard method

<p>Claims of 5806063</p> <p>1. A method of processing symbolic representations of dates stored in a database, comprising the steps of</p> <p>providing a database with symbolic representations of dates stored therein according to a format wherein $M_1 M_2$ is the numerical month designator, $D_1 D_2$ is the numerical day designator, and $Y_1 Y_2$ is the numerical year designator, all of the symbolic representations of dates falling within a 10-decade period of time;</p>	<p>Shaughnessy Patent 5,630,118</p> <p>The disclosed software assigns a century value to a two digit year date (7/6+), which is processing of symbolic dates</p> <p>One format which can form an input is YYMMDD, see Date Type "B" in the appendix at col.18, formats such as B have a 100 year (10-decade) cycle, see the footnote defining C1, in other words the format wraps modulo 100</p>
<p>selecting a 10-decade window with a $Y_A Y_B$ value for the first decade of the window, $Y_A Y_B$ being no later than the earliest $Y_1 Y_2$ year designator in the database;</p>	<p>software "determine[s] end of current 100 year cycle", step 16, fig. 2, 3 or 4, as the "end" of the 100 year range, the "end" year is one less than the beginning (if "37" is the last year of a 100 year period, "38" is the first year of the same period), the "end" year is no later than any date in the data base of 100 year cycle data</p>
<p>determining a century designator $C_1 C_2$ for each symbolic representation of a date in the database, $C_1 C_2$ having a first value if $Y_1 Y_2$ is less than $Y_A Y_B$ and having a second value if $Y_1 Y_2$ is equal to or greater than $Y_A Y_B$, and</p>	<p>the century designator is determined by comparing two digit representation to the end of the 100 year cycle date, if the year being processed is greater, then the earlier century value is assigned and vice versa; (col. 7, lines 5-15)</p>
<p>reformatting the symbolic representation of the date with the values $C_1 C_2$, $Y_1 Y_2$, $M_1 M_2$, and $D_1 D_2$ to facilitate further processing of the dates.</p>	<p>the reformatting is described at 6/57-65</p>
<p>2. The method of claim 1, wherein the 10-decade window includes the decade beginning in the year 2000.</p>	<p>Since the patent is directed to Y2K, it by definition includes the year 2000, col. 8, lines 7-18 refer to the 20th and 21st centuries</p>
<p>3. The method of claim 2, wherein the step of determining includes the step of</p>	

4. The method of claim 1, including an additional step, after the step of reformatting, of	
sorting the symbolic representations of dates.	Shaughnessy refers (col. 1, line 26) to date comparisons; sorting is merely sequential date comparisons
5. The method of claim 1, wherein the step of reformatting includes the step of reformatting each symbolic representation of a date into the format C ₁ C ₂ Y ₁ Y ₂ M ₁ M ₂ D ₁ D ₂ .	See 6/58-65, 8/10-20
6. The method of claim 5, including an additional step, after the step of reformatting, of	
sorting the symbolic representations of dates using a numerical-order sort	Shaughnessy refers (col. 1, line 26 and column 8, line 35 – column 12, line 20) to date comparisons; sorting is merely sequential date comparisons
7. The method of claim 1, wherein the step of providing a database includes the step of	
converting pre-existing date information having a different format into the format wherein M ₁ M ₂ is the numerical month designator, D ₁ D ₂ is the numerical day designator and Y ₁ Y ₂ is the numerical year designator.	see the appendix at col. 18 to illustrate a host of formats and col. 6, lines 58-65

00005628-020200

11. A method of processing dates in a database, comprising the steps of	The disclosed software assigns a century value to a two digit year date (7/6+) which is processing of dates
providing a database with symbolic representations of dates stored therein according to a format wherein $M_1 M_2$ is the numerical month designator, $D_1 D_2$ is the numerical day designator, and $Y_1 Y_2$ is the numerical year designator, all of dates falling within a 10-decade period of time which includes the decade beginning in the year 2000;	One format which can form an input is YYMMDD, see Date Type "B" in the appendix at col.18, formats such as B have a 100 year (10-decade) cycle, see the footnote defining C1, in other words the format wraps modulo 100
selecting a 10-decade window with a $Y_A Y_B$ value for the first decade of the window, $Y_A Y_B$ being no later than the earliest $Y_1 Y_2$ year designator in the database;	software "determine[s] end of current 100 year cycle", step 16, fig. 2, 3 or 4, as the "end" of the 100 year range, the "end" year is one less than the beginning (if "37" is the last year of a 100 year period, "38" is the first year of the same period), the "end" year is no later than any date in the data base of 100 year cycle data
determining a century designator $C_1 C_2$ for each symbolic representation of a date in the database, $C_1 C_2$ having a first value if $Y_1 Y_2$ is less than $Y_A Y_B$ and having a second value if $Y_1 Y_2$ is equal to or greater than $Y_A Y_B$;	the century designator is determined by comparing two digit representation to the end of the 100 year cycle date, if the year being processed is greater, then the earlier century value is assigned and vice versa; (col. 7, lines 5-15)
reformatting each date in the form $C_1 C_2 Y_1 Y_2 M_1 M_2 D_1 D_2$ to facilitate further processing of the dates; and	the reformatting is described at 6/57-65
sorting the dates in the form $C_1 C_2 Y_1 Y_2 M_1 M_2 D_1 D_2$.	Shaughnessy refers at col. 1, line 26 and in columns 8-10, to date comparisons, sorting is merely sequential date comparisons
12. The method of claim 11, wherein the step of providing a database includes the step of	
converting pre-existing date information having a different format into the format wherein $M_1 M_2$ is the numerical month designator, $D_1 D_2$ is the numerical day designator, and $Y_1 Y_2$ is the numerical year designator.	see the appendix at col. 18 to illustrate a host of formats and col. 6, lines 58-65

Claims of 5806063	Browe
1. A method of processing symbolic representations of dates stored in a database, comprising the steps of	Browe describes a utility for date calculations
providing a database with symbolic representations of dates stored therein according to a format wherein M ₁ M ₂ is the numerical month designator, D ₁ D ₂ is the numerical day designator, and Y ₁ Y ₂ is the numerical year designator, all of the symbolic representations of dates falling within a 10-decade period of time;	the parameters of the data base represented on p. 73 include MMDDYY, the comment (p.70) states "The reference date is typed in MMDDYY format and must refer to a date from 01/01/1950 to 12/31/2049 for this focexec to work correctly" That is the 100 year range (10-decades)
selecting a 10-decade window with a Y _A Y _B value for the first decade of the window, Y _A Y _B being no later than the earliest Y ₁ Y ₂ year designator in the database;	on p. 71, see the line of code: "If &YR GE 50 and &YR LE 99 then 19 &YR ELSE 20 &YR" In this expression Y _A Y _B =50
determining a century designator C ₁ C ₂ for each symbolic representation of a date in the database, C ₁ C ₂ having a first value if Y ₁ Y ₂ is less than Y _A Y _B and having a second value if Y ₁ Y ₂ is equal to or greater than Y _A Y _B , and	on p. 71, see the line of code: "If &YR GE 50 and &YR LE 99 then 19 &YR ELSE 20 &YR" In this expression "19" and "20" are the values for , C ₁ C ₂ ; the logic is the same as the claimed logic except stated in inverse order
reformatting the symbolic representation of the date with the values C ₁ C ₂ , Y ₁ Y ₂ , M ₁ M ₂ , and D ₁ D ₂ to facilitate further processing of the dates.	on p. 71, see the line of code: "If &YR GE 50 and &YR LE 99 then 19 &YR ELSE 20 &YR". In this expression the "19 &YR" and "20 &YR" show the reformatting with the determined century designator
2. The method of claim 1, wherein the 10-decade window includes the decade beginning in the year 2000.	The period 01/01/1950 to 12/31/2049 includes the decade beginning in the year 2000.

2

Claims of 5806063	Ohms in light of (35USC 103) JP 05-027947 ('947)
4. The method of claim 1, including an additional step, after the step of reformatting, of	The manner in which Ohms anticipates the terms of claim 1 has already been described.
sorting the symbolic representations of dates.	'947 describes under "Constitution" that the processing to correct the Y2K problem is implemented "before sort/merge processing". This teaches that sorting can be implemented after Y2K correction. Since both Ohms and '947 are directed to Y2K correction, those skilled in the art would view both teachings as applying to the same problem. If needed then, '947's "sorting" statement reveals that sorting may be implemented after Y2K correction.
11. A method of processing dates in a database, comprising the steps of	Both Ohms and '947 are directed to date processing
providing a database with symbolic representations of dates stored therein according to a format wherein M ₁ M ₂ is the numerical month designator, D ₁ D ₂ is the numerical day designator, and Y ₁ Y ₂ is the numerical year designator, all of dates falling within a 10-decade period of time which includes the decade beginning in the year 2000;	see the table on p; 247 of Ohms including reference to "a short Gregorian date", the conversion function works with a number of data formats including this one, Ohms describes, at p. 249, the 100 year limitation (identical to 10-decades)
selecting a 10-decade window with a Y _A Y _B value for the first decade of the window, Y _A Y _B being no later than the earliest Y ₁ Y ₂ year designator in the database;	see Ohms, p. 248, right hand column – specify a year as the desired starting point of the range – this is Y _A Y _B , and it is no later than any year in the data base
determining a century designator C ₁ C ₂ for each symbolic representation of a date in the database, C ₁ C ₂ having a first value if Y ₁ Y ₂ is less than Y _A Y _B and having a second value if Y ₁ Y ₂ is equal to or greater than Y _A Y _B ;	in Ohms, the century designation is determined by comparing the year date with the desired starting point, if the year in question is greater then the century is the earlier one and vice versa, see p. 248
reformatting each date in the form C ₁ C ₂ Y ₁ Y ₂ M ₁ M ₂ D ₁ D ₂ to facilitate further processing of the dates; and	in Ohms, the "implied century" (see p. 248, right hand column) which is the result of the date "conversion" is the reformatted date, '947, on the other hand replaces the code of 21 st century years to maintain correct date order[0015], at [0020]

sorting the dates in the form C ₁ C ₂ Y ₁ Y ₂ M ₁ M ₂ D ₁ D ₂ .	'947 describes under "Constitution" that the processing to correct the Y2K problem is implemented "before sort/merge processing". This teaches that sorting can be implemented after Y2K correction.
	Since both Ohms and '947 are directed to Y2K, it would be natural for those skilled in the art to view the two teachings together, including modifying Ohms' formatting as well as using the '947 suggestion concerning sorting.
13. The method of claim 11, wherein the step of selecting includes the step of	The application of Ohms and '947 to claim 11 is described above.
selecting Y _A Y _B such that Y _B is 0 (zero).	'947 has an example of the use of "1960" as the earliest date in the range [0011]. This results in dates of 00-59 being determined to be in the 21 st century, i.e., the '947 parameter corresponding to Y _A Y _B is "60".
14. The method of claim 11, including an additional step, after the step of sorting, of	The application of Ohms and '947 to claim 11 is described above,
storing the sorted dates and their associated information back into the database.	Ohms, at p. 248-249 discusses the storage issue and expressly mentions storing date data with four digit year representations, i.e., just the format of the claimed C ₁ C ₂ Y ₁ Y ₂ M ₁ M ₂ D ₁ D ₂ .
15. The method of claim 14, including the additional step, after the step of sorting, of	
manipulating information in the database having the reformatted date therein.	The application of Ohms and '947 to claim 14 is described above. Ohms refers to the need for date conversions for handling future applications (p. 250). Future applications for data base programs require manipulation of the data

0000528-00200

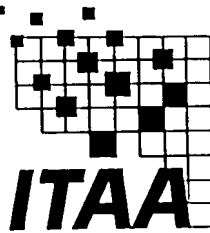
Claims of 5806063	Shaughnessy Patent 5630118 in light of (35USC 103) JP 05-027947 ('947)
4. The method of claim 1, including an additional step, after the step of reformatting, of	The manner in which Shaughnessy patent 5630118 applies to claim 1 has already been described.
sorting the symbolic representations of dates.	'947 describes, under "Constitution" that processing to correct the Y2K problem is implemented "before sort/merge processing". This teaches that sorting can be implemented after Y2K correction. Since both Shaughnessy and '947 are directed to Y2K correction, those skilled in the art would view both teachings as applying to the same problem. If needed then, '947's "sorting" statement reveals that sorting may be implemented after Y2K correction.
6. The method of claim 5, including an additional step, after the step of reformatting, of	The manner in which Shaughnessy patent 5630118 applies to claim 5 has already been described.
sorting the symbolic representations of dates using a numerical-order sort.	'947 describes under "Constitution" that the processing to correct the Y2K problem is implemented "before sort/merge processing". This teaches that sorting can be implemented after Y2K correction. Since both Shaughnessy and '947 are directed to Y2K correction, those skilled in the art would view both teachings as applying to the same problem. If needed then, '947's "sorting" statement reveals that sorting may be implemented after Y2K correction.
8. The method of claim 1, wherein the step of selecting includes the step of	The manner in which Shaughnessy patent 5630118 applies to claim 1 has already been described.
selecting $Y_A Y_B$ such that Y_B is 0 (zero).	'947 has an example in which the 'earliest date' in the data base is "60", this is an example of $Y_A Y_B$ such that Y_B is 0 (zero). Since both Shaughnessy and '947 are directed to Y2K correction, those skilled in the art

	would view both teachings as applying to the same problem and
	rely on the '947 suggestion of selecting a parameter corresponding to $Y_A Y_B$ such that Y_B is 0 (zero), as described in '947 if any such suggestion might be necessary.
13. The method of claim 11, wherein the step of selecting includes the step of	The manner in which Shaughnessy patent 5630118 applies to claim 11 has already been described.
selecting $Y_A Y_B$ such that Y_B is 0 (zero).	'947 has an example in which the 'earliest date' in the data base is "60", this is an example of $Y_A Y_B$ such that Y_B is 0 (zero). Since both Shaughnessy and '947 are directed to Y2K correction, those skilled in the art would view both teachings as applying to the same problem and rely on the '947 suggestion of selecting a parameter corresponding to $Y_A Y_B$ such that Y_B is 0 (zero), as described in '947 if any such suggestion might be necessary.

00005628-020200

Claims of 5806063	Ohms considered with The Millennium Journal (35 USC 103)
5. The method of claim 1, wherein the step of reformatting includes the step of	The application of Ohms to claim 1 has already been described
reformatting each symbolic representation of a date into the format C ₁ C ₂ Y ₁ Y ₂ M ₁ M ₂ D ₁ D ₂ .	The Millennium Journal, in the first row of the Table on page 4 under the heading "Satisfying the Standard" show this particular format. It has already been noted that Ohms describes using the YY, MM and DD parameters as well as determining the CC data. Given that Ohms and The Millennium Journal are both directed to Y2K, that the latter reports on the use of this format and that Ohms describes the presence of just this data, it would have been obvious at the time to follow the report in The Millennium Journal and use this format.
11. A method of processing dates in a database, comprising the steps of	Ohms and The Millennium Journal both describe Y2K related date processing
providing a database with symbolic representations of dates stored therein according to a format wherein M ₁ M ₂ is the numerical month designator, D ₁ D ₂ is the numerical day designator, and Y ₁ Y ₂ is the numerical year designator, all of dates falling within a 10-decade period of time which includes the decade beginning in the year 2000;	see the table on p; 247 including reference to "a short Gregorian date", the conversion function works with a number of data formats including this one, Ohms describes, at p. 249, the 100 year limitation (identical to 10-decades)
selecting a 10-decade window with a Y _A Y _B value for the first decade of the window, Y _A Y _B being no later than the earliest Y ₁ Y ₂ year designator in the database;	see p. 248, right hand column - specify a year as the desired starting point of the range - this is Y _A Y _B , and it is no later than any year in the data base
determining a century designator C ₁ C ₂ for each symbolic representation of a date in the database, C ₁ C ₂ having a first value if Y ₁ Y ₂ is less than Y _A Y _B and having a	the century designation is determined by comparing the year date with the desired starting point, if the year in question is greater then the century is the earlier one and vice

2
22,



Commissioner Dickinson
ASSISTANT SECRETARY
AND COMMISSIONER

2000 FEB 15 PM 3:42

U.S. PATENT
AND
TRADEMARK OFFICE

*cc: Ret'g
A/c Pat's*

February 8, 2000

The Honorable Q. Todd Dickinson
Assistant Secretary and Commissioner of Patents & Trademarks
United States Patent and Trademark Office
906 CPK2
Crystal Park, VA 22202

Dear Mr. Commissioner,

The Information Technology Association of America (ITAA) is the leading national trade association of the enterprise software, telecommunications, Internet, ASP, systems integration and IT services industry. ITAA consists of 400 direct and 26,000 affiliate corporate members throughout the U.S., and coordinates a global network of 39 countries' IT associations. The Association plays the leading role in issues of IT industry concern including taxes and finance policy, intellectual property, telecommunications competition, workforce and education, encryption, critical infrastructure protection, online privacy and consumer protection, securities litigation reform, government IT procurement, and human resources policy. ITAA members range from the smallest IT start-ups to industry leaders.

Over the past four years ITAA has been the leading voice in bringing marketplace awareness and offering strategic solutions to prepare for and successfully confront the Y2K software conversion issue. We have worked closely with a number of vertical industry associations, ITAA members' customers, government officials, and the public-at-large in publicizing, speaking out on, and tackling the many complex public policy issues surrounding the century date change.

ITAA heard about the patent that had been granted to McDonnell Douglas (later, Boeing) and 'assigned back' to the 'inventor' Bruce Dickens in the Fall of 1999. We, subsequently, learned that Mr. Dickens had sent form letters to a number of ITAA members and their customers regarding the possible need to enter into a licensing arrangement with him with respect to their Y2K remediation efforts. Without taking any position regarding the validity of the patent, or whether or not ITAA members had used a remediation method in violation of the Dickens' patent, ITAA began monitoring developments on our Internet website in October of 1999 and, then started receiving a number of potential "prior art" citations from members and interested parties around the country. See <http://www.ita.org/year2000/dickens.htm>.

Information Technology Association of America

1616 N. Fort Myer Drive, Suite 1300, Arlington, Virginia 22209-3106 ■ Phone: (703) 522-5055 Fax: (703) 525-2279

We learned on December 21, 1999 of the USPTO's decision to order a reexamination of the Dickens' patent. While the Ohm's article has been cited as one of many potential prior art examples, ITAA wishes to share with the USPTO other numerous citations that may be relevant to the reexamination process. ITAA and its members realize that the reexamination order does not mean that the patent in question (or any 'windowing' patent that may have been issued even before the Dickens patent) will be invalidated. We have, in fact, on our website continued to encourage companies to carry out their own internal analysis as to whether or not they 'may' have utilized techniques and methods claimed by Mr. Dickens. At the same time, we have also continued to receive even more potential prior art citations.

Currently listed on ITAA's Internet website – are forty-seven (47) potential prior art citations. We bring these citations to your attention with the hope that they may be helpful in the reexamination process. If and when further citations are received, ITAA plans to post them on our website.

1. ANSI X3.226-1994, "Information Technology – Programming Language – Common Lisp", 1994, Section 25.1.4.1.
2. Guy Steel, Common Lisp, The Language, Second Edition, Digital Equipment Corporation, 1990, p.702.
3. Guy Steel, Common Lisp, The Language, Digital Equipment Corporation, 1984.
4. David Moon, etc. al., Lisp Machine Manual, Sixth Edition, MIT Artificial Intelligence Laboratory, 1984, p.776. Note: The Compatibility Note in "Common Lisp, The Language" refers to an earlier edition of that language, prior to the introduction of the rolling "10-decade" window. The fourth edition of the Lisp Machine Manual, for instance, does not include the feature.
5. Oracle Corporation, SQL Language Reference Manual, Version 7.0, Part number 778-70-1292, December 1992, pp. 2-21, 2-27 to 2-31, 3-30 to 3-59.
6. International Standard ISO 8601: 1988 (E), "Data Elements and Interchange Formats – Information Exchange – Representation of Dates and Times," International Organization for Standardization, 1st edition, 1988-06-15.
7. International Standard ISO 2014: 1976 (E), "Writing of Calendar Dates in All-Numeric Form", International Organization for Standardization, first edition 1976-04-01.
8. Jonathan Postel, Internet RFC 753, "Internet Message Protocol", March 1979.
9. Richard Bergeon, The Millennium Journal, Vol. II.IV, July 1995, pp. 2-4.
10. IBM SAA AD/Cycle Language Environment/370 Programming Guide Version 1 Release 1, December 1991, pp. 360-61.
11. Gary Browe, "Intelligent Report Maintenance Using Dialogue Manager", Focus Systems Journal, March 1990, pp. 70-71.
12. B. G. Ohms, "Computer Processing of Dates Outside the Twentieth Century", IBM Systems Journal, Vol. 25, No. 2, 1986, pp. 244-50.
13. Oracle Manual, Oracle Developer/2000 Forms4.5 Reference Manual Volume 2, Part No. A32510-2 Copyright Oracle Corporation 1994; pp 5-91.
14. Sybase, Inc., Sybase SQL Server Reference Manual, Release 10.0, June 1994, pp. 2-9-12, 2-15, 2-19-23, 3/34-38.

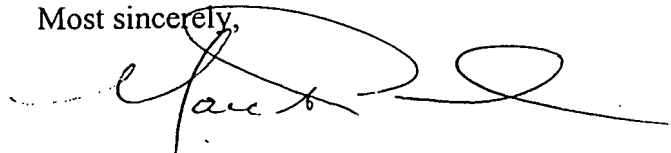
15. Sybase, Inc., Commands Reference Manual for SYBASE SQL Server for UNIX, SYBASE SQL Server Release 4.9.1, Document ID: 32270-01-0491-01, Change 2, Oct 1, 1992.
16. Oracle Corporation, Oracle7 Server SQL Reference, Release 7.2, April 1995, Part No. A20325-2, 1995, pp. 3-66, 3-67.
17. SAS Institute, Inc., SAS Language: Reference, Version 6, First Edition, Cary, NC; SAS Institute Inc., 1990, pp. 63, 68, 85, 182, 536-539, 649-655, 670, 682-685, 690-695, 698-699, 704-706, 746, 790-791.
18. Stan Milam, "Extended Data Library for C", C-C++ Users Journal, Vol. 12, No. 10, October 1994, pp. 67-80.
19. Stephen J. Straley, Programming in Clipper, The Definitive Guide to the Clipper dBASE Compiler, The Second Edition, Addison-Wesley Publishing Company, September 1988.
20. Nantucket Corporation, Clipper 5.0 Reference Manual, 1990.
21. David E. Whitney, "Year 2000", Presented at SHARE 84 in Los Angeles, CA, Feb 26-Mar 3, 1995 and SHARE 85 in Orlando, FL, Aug 13-18, 1995.
22. "Here's One Alternative Date Method", Tick, Tick, Tick, Vol. 1, No. 1, 1993, pg.1 and "One Man's Opinions," p.5.
23. Roger Knights – (a) COBOL Committee Working Paper, "Turn-of-the-Century Problem", CC #83118, August 23, 1983. (b) COBOL Committee Working Paper, "Date-Handling Functions", CC #84051, October 2, 1984. (c) Public Review Comments incorporated in pp 46-48 of document J4/97-0318, "J4 Response to U.S. Public Review Comments on COBOL WD 1.4", November 5, 1997.
24. Anon, Hewlett Packard, "Procedure Division in the Nucleus", Sept. 1998.
25. Anon, Hewlett Packard, "Conversion Functions", March 1987.
26. Anon, IBM, "Date Adjustment At The Turn Of The Century", TDB, May 1986, N265 05-6 [Freeman].
27. Anon, IBM, DFSORT – Application Programming Guide, Release 11.1, Feb. 1991.
28. Anon, IBM, "Programming Reference", Language Environment for MVS & VM (Sept. 1995).
29. Anon, IBM, "The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation", 1995, 1996.
30. Arnold, "Resolving Year 2000 Problems in Legacy Software", Software Quality Week, June 1, 1995, SF.
31. DeJager, "Doomsday 2000", Computerworld, Sept. 6, 1993.
32. Furman, "Party When It's 1999", Software Magazine, April 1995, p.6.
33. Gasteiger, D., "Make a Date With 1-2-3 ...", Lotus, v 7, #4, p. 24, March 1991.
34. Gillin, "A Y2K Pioneer Seeks (And Deserves) Recognition", 1984, Schoen.
35. Glass, Robert, "The Next Date Crisis and the Ones After That", Comm of the ACM, v 40, #1, Jan 1997.
36. Greer, "Method of Sorting Dates and Time Allowing for Wrapping", IBM TDB, v 37, #8, August 1994, pp. 381-2.
37. Hart et al, "A Scaleable, Automated Process for Year 2000 System Correction", PROC 18th International Conference on Software Engineering, 25-30 Mar. 1996, pp 475-484, Mar 30, 1996.

38. Hayes, "Waiting for 01-01-00", American Scientist, v 83, #1, Jan. 1995.
39. Matthews, "Excel 4 for Windows", 1992, pp. 347.
40. Meyer, "Julian and Gregorian Calendars", Dr. Dobbs Journal, March 1993.
41. Neuhaus, "Databases", PC Magazine, v 9, #16, p. 471, Sept. 25, 1990.
42. Pieptea, D.R., "What Will The Change Of The Millennium Do To Our Data Processing", Management Information Systems Quarterly, v 10, #2, pp. 103-4, June 1, 1986.
43. Roberts, "DataServer 2000: Computer Software Prepares Legacy Systems for Year 2000", Enterprise Systems Journal, Nov. 1994, p. 10.
44. Schoen, "The Charmar Correction", copyright registration TX 144-098, Dec. 1983.
45. Smith, H. J., "What's Ahead For 2000 AD?," APL Quote Quad, v 20, #4, p. 364, July 1990.
46. Vandercook, "Now Is The Time", Systems Management, March 1995.
47. Xenakis, "Preparing For 2000", Information Week, Feb. 26, 1990.

In addition to the potential prior art citations listed above, we also wish to point out that a number of questions have come up regarding some previously issued patents covering the windowing technique in light of the Dickens claim. We hope that your examination will also look into any possible conflicts surrounding these patents, as well.

We hope this information will be of assistance in the reexamination process.

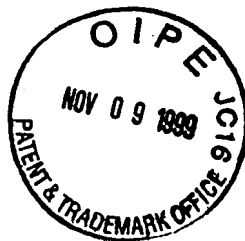
Most sincerely,



Marc A. Pearl
General Counsel & Senior Vice President

November 4, 1999

Assistant Commissioner for Patents
Washington, DC 20231



BD of Apple

DAC

LOC-

Dear Sir,

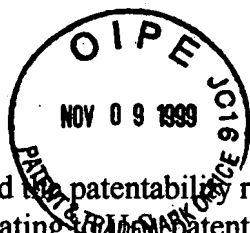
Enclosed please find two copies of documentation that I am submitting under 37 CFR 1.501 that I believe is relevant to the patentability of U.S. patent #5,806,063 issued on September 8, 1998 to Bruce Dickens. This documentation consist of a description of the printed materials and their relevance to the claims of the patent and a list of nine citations to prior art. Also attached are copies of form PTO/SB/42 with a subset of those citations listed. Completing the documentation are copies of pages from the listed citations illustrating the prior art.

In accordance with 37 CFR 1.501, I wish this material to be made a part of the file on patent #5,806,063, so that these references to prior art may be made available to anyone investigating this patent and its patentability. I would prefer that this cover letter not be made a part of the file, however.

Frankly, this appears to be yet another software patent that should never have been awarded. The techniques "taught" by the patent have been in common use for at least fifteen years — at least twelve years prior to the filing of the patent application. Several of the claims are not only obvious to any competent software engineer, but, where they are applicable, are demanded by the application, and are not a matter of choice.

Sincerely,

Patrick A. O'Donnell
16 Margaret Ln
Billerica, MA 02139



Under the authority of 37 CFR 1.501 and the patentability requirements of 35 USC 102(a) and (b) I wish to cite prior art relating to the patent number 5,806,063, "Date formatting and sorting for dates spanning the turn of the century" issued on September 8, 1998 to Bruce Dickens.

I wish to bring to the attention of the Patent Office the below cited prior art which I believe bears on the claims of the patent. Pursuant to 35 USC 102(a) and (b), some of the methods claimed in the patent were known and used by others and described in printed publications in this and foreign countries more than one year prior to the date of the application for patent. In particular, the method of claim 1, where dates are represented symbolically with a two-digit year and interpreted with respect to a "10-decade window", has been used for at least 15 years. The method is mandated in the ANSI Common Lisp standard¹, as described in section 25.1.4.1, "Decoded Time", of that standard. That section has a long history; the technique is also described in *Common Lisp, The Language, Second Edition*.² (It is also described in the first edition of that work.³) The same method is described differently in the *Lisp Machine Manual*, chapter 36, "Dates and Times".⁴ The application to interpretation of dates stored in databases is immediate — indeed one would have to go out of their way to avoid it if the database application were written in Common Lisp. Further, at least two database vendors have, at least four years prior to the filing of this patent application, defined this method for interpreting two-digit year representations.^{5,6}

Secondly, with respect to dependent claim 5 and the final step of claim 1 (which, not incidentally, appear identical), the described date format is previously described in International Standard ISO 8601 : 1988 (E)⁷, and also in the earlier ISO 2014-1976 (E)⁸, which the former supercedes. The usefulness of this date format for automated manipulation is expressly mentioned in the ISO standards; the application to sorting (dependent claims 4 and 6 of the patent) is immediately obvious. The ISO standards are also described and referenced in Internet RFC 753⁹, among others.

¹ ANSI X3.226-1994, "Information Technology — Programming Language — Common Lisp", 1994. Section 25.1.4.1.

² Guy Steel, *Common Lisp, The Language, Second Edition*, Digital Equipment Corporation, 1990, p.702.

³ Guy Steel, *Common Lisp, The Language*, Digital Equipment Corporation, 1984.

⁴ David Moon, et. al., *Lisp Machine Manual*, Sixth Edition, MIT Artificial Intelligence Laboratory, 1984, p. 776. Note: the "Compatibility Note" in *Common Lisp, The Language* referring to Lisp Machine Lisp was referring to an earlier edition of that language, prior to the introduction of the rolling "10-decade" window. The fourth edition of the *Lisp Machine Manual*, for instance, does not include the feature.

⁵ *Commands Reference Manual for Sybase SQL Server for UNIX*, Release 4.9.1., Document ID 32270-01-0491-01, 1 October 1992. Chapter 2, Section "Date Functions", p 2-106.

⁶ Oracle Corporation, *SQL Language Reference Manual*, Version 7.0, Part number 778-70-0292, May 1992. p. 4-49.

⁷ *International Standard ISO 8601 : 1988 (E)*, "Data elements and interchange formats — Information exchange — Representation of dates and times", International Organization for Standardization, first edition 1988-06-15.

⁸ *International Standard ISO 2014 : 1976 (E)*, "Writing of calendar dates in all-numeric form", International Organization for Standardization, first edition 1976-04-01.

⁹ Jonathan Postel, Internet RFC 753, "Internet Message Protocol", March 1979.

Sheet 2 of 2

Burden Hour Statement: This form is estimated to take 2.0 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231. **DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Washington, DC 20231.**

Sheet 1 of 2

37 CFR 1.501 INFORMATION DISCLOSURE CITATION IN A PATENT (Use several sheets if necessary)						Docket Number (Optional)		Patent Number 5,806,063		
						Applicant Bruce Dickens				
						Issue Date Sep 8, 1998		Group Art Unit		
U. S. PATENT DOCUMENTS										
EXAMINER INITIAL	DOCUMENT NUMBER				DATE	NAME		CLASS	SUBCLASS	FILING DATE IF APPROPRIATE
FOREIGN PATENT DOCUMENTS										
	DOCUMENT NUMBER				DATE	COUNTRY	CLASS	SUBCLASS	Translation YES NO	
OTHER DOCUMENTS (Including Author, Title, Date, Pertinent Pages, Etc.)										
	ANSI X3.226-1994, Programming Language Common Lisp, 1994 Section 25.1.4.1									
	Guy Steel, Common Lisp, The Language, Second Edition, 1990, p 702									
	Commands Reference Manual for SQL Server, Release 4.9.1 Sybase Corporation, October 1, 1992, Chapter 2, Section "Date Functions"									
EXAMINER						DATE CONSIDERED				

Burden Hour Statement: This form is estimated to take 2.0 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231. **DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO:** Assistant Commissioner for Patents, Washington, DC 20231.

5446 45th AV SW
Seattle WA 98136
206-932-9323
Dec. 2, 1999

Gerald Goldberg, Director
U.S. Patent & Trademark Office
Technology Center 2700
Washington, DC 20231
Dec. 1, 1999

Received
DEC 8 1999
Directors Office
Group 2700

Dear Mr. Goldberg:

I spoke to you last week after reading a Nov. 12 AP story about Bruce Dickens' 1998 patent (#5,806,063) on turn-of-the-century date-handling, which I believe I anticipated in two working papers I submitted to the CODASYL COBOL Committee (CCC) in 1983 & 1984. As per our conversation, I'm enclosing those papers, plus—for the sake of completeness—a third paper I submitted to the ANSI COBOL Committee (X3J4) in 1996. These papers are:

1. 83-8-23 (83118) RK-WP Turn-of-the-Century Problem; plus p. 36 of the Nov. 1983 CCC minutes. (The quote at the start of the paper comes from p. 93 of, I think, Computing in the 1980s, published in 1976 by SHARE, the IBM user group.)
2. 84-10-2 (84051) RK-WP Date-Handling Functions (relevant parts are on pp. 3-4 and Item 3 of the supplement); plus p. 16 of the Nov. 1984 minutes.
3. 96-11-20 (96-0478) Public Review Comments from Roger Knights; incorporated in pp. 46-48 of document J4/97-0318, "J4 Response to U.S. Public Review Comments on COBOL WD 1.4".

Official copies of these documents can be obtained from the archivists for the CCC and X3J4 groups respectively, namely:

Don Nelson
Compaq Computers CAC01-02
19333 Valico Pkwy.
Cupertino, CA 95014-2599
408-285-5203 (fax -5245)

John Brieschke
Unisys Corp. MS 4872
POB 64942
St. Paul, MN 55164
651-635-5291 (fax -5088)

Please send me the form for filing a petition for a patent reexamination.

You mentioned that firms from whom Mr. Dickens has requested royalties might be willing to repay me the non-trivial cost of filing such a petition. (I assume that vendors of COBOL compilers would be the most interested.) You may forward my contact information to them, as well as copies of this letter and enclosures.

I suspect my papers may pre-empt earlier patents on this topic, as they cover not only the expansion (to 8 digits) and internal rearrangement of "windowed" dates (the focus of Dickens' patent), but the basic technique of windowing itself. I suspect they may also pre-empt certain pending patents dealing with other techniques I described, namely:

1. "Sliding windows", where the pivot year (which I called the VTOC or VEOC) is not a fixed value but an offset from whatever the current year is. (To avoid having to manually reset the pivot year as time passes.)
2. A run-time facility to disable windowing when the current year is not in a date range near the turn of the century (for the sake of speed). E.g., END CENTURY AT CURRENT-YEAR +5 WHEN CURRENT-YEAR IS 95 THRU 04 would disable windowing prior to 1995 and after 2004.
3. A compile-time facility to disable production of window-checking code when the current year is not near the turn of the century (for the sake of speed and space). E.g., a compiler directive might state, WITH WINDOWING UNLESS DATE < 19900000. (This directive might be global or local.)
4. A compiler facility to modify CURRENT-DATE (to obtain desired results during reruns or testing). (CURRENT-DATE is a COBOL "special register" containing the run-time date; other languages have a similar data item.)

Since I submitted over 100 papers for the CCC, I wonder if I might have anticipated additional patents, including some on which compiler vendors are paying royalties.

It's a pity the Software Patent Institute at the Univ. of Michigan (313-769-4083) hasn't received the funding it needs to operate effectively, as that would have enabled your examiner to discover my prior art.

Yours truly,

A handwritten signature in black ink that reads "Roger Knights". The signature is written in a cursive, flowing style with a large, stylized 'R' and 'K'.

(Roger Knights)

EAST - [08459040.wsp:1]

File View Edit Tools Window Help

☐ Drafts
☐ Pending
☒ Active

- L1: (30) (COMPLEMENTING-OR COM
- L2: (10591) (vaccinia or pox\$ or \$pox
- L3: (390562) essential
- L4: (5306) non adj essential
- L5: (195) 2 with 3
- L6: (129) 2 with 4
- L7: (66) 5 not 6
- L12: (195) (2 with 3).clm
- L13: (36) (2 with 3).clm.

☐ Failed
☒ Saved

(27511) virus

(21221) COMPLEMENTING-OR COM

DBs: USPAT

Default operator: OR

☐ Plurals
☒ Highlight all hit terms initially

	Type	L #	Hits	Search Text	DBs	Time Stamp	Co	Error Definition	Err
1	BRS	L1	30	(COMPLEMENTING OR COMPLEMENTED OR COMPLEMENTS) SAME (vaccinia or pox\$ or \$pox or \$poxvirus)	USPAT	2002/05/13 11:17			0
2	BRS	L2	10591	(vaccinia or pox\$ or \$pox or \$poxvirus)	USPAT	2002/05/13 11:17			0
3	BRS	L3	390562	essential	USPAT	2002/05/13 11:17			0
4	BRS	L4	5306	non adj essential	USPAT	2002/05/13 11:18			0
5	BRS	L5	195	2 with 3	USPAT	2002/05/13 11:19			0
6	BRS	L6	129	2 with 4	USPAT	2002/05/13 11:18			0
7	BRS	L7	66	5 not 6	USPAT	2002/05/13 11:18			0
8	BRS	L12	195	(2 with 3).clm	USPAT	2002/05/13 11:20			0
9	BRS	L13	36	(2 with 3).clm.	USPAT	2002/05/13 11:20			0

COBOL COMMITTEE
WORKING PAPER

C.C.# : 83118

ORG.# : RK-WP

AUTHOR: Roger Knights

DATE : Aug. 23, 1983

SUBJECT: Turn-of-the-Century Problem

Here is a scenario to be avoided:

The 2000 phenomenon

In the last weeks of the year 1999 and in early 2000, business in the computerized economies of the world all but collapsed. For bills rendered in 1999 but due in 2000, overdue notices were issued already in 1999, charging the debtor interest for some 99 years. After January 1, 2000, many systems failed to issue overdue notices for amounts due in 1999 but not yet paid. The data in many reports were printed in the incorrect sequence: data pertaining to the year 2000 preceded data for the year 1999 instead of following it. Many computer runs terminated abnormally during this time because of overflow and similar errors. For software maintenance personnel this was a very difficult and trying time, during which the time pressure under which they worked was unusually severe.

The problems could all be attributed to the widespread use of a two-digit field for the year in computerized data files. In the late 1970's, the 1980's and the early 1990's, programmers had made incorrect assumptions regarding the operational lifetimes of the programs and the data files they designed.

It would seem desirable to add a clause that would shift the effective turn of the century far enough forward that a comparison or arithmetic operation between dates in different centuries could never be attempted. Let's say that a programmer codes "END CENTURY AT 09" in the entry for all two-digit year-fields in his program. 09 is ten years after 99. Therefore, prior to all comparisons or arithmetic operations involving that field, it will be moved to a hidden work area and ten will be subtracted from it, & 100 will be added if negative-result; then the operation will be executed. If the hidden field is the target of a math operation, it will have ten added back to it (100's truncated) after the operation, and then it will be moved back to the original source field. In effect, years are thus left-rotated by ten, temporarily.

As long as the years represented in a user's date fields are less than 100 years apart, it will be possible, using this proposed clause, for all later years to be effectively higher than all earlier years, regardless of the century they fall in. Note that if it is desired to allow a comparison of dates to be accurate for the maximum range (99 years), it would be necessary to update the END CENTURY operand every year. This is undesirable, and could be automated by allowing the operand to take the form shown here:

END CENTURY AT CURRENT-YEAR + 1

Where CURRENT-YEAR contains the leftmost 2 digits of DATE.

During any voluminous sorting operation, the inefficiency of this left-rotating would be very apparent; in fact, it would be desirable to avoid this inefficiency wherever possible. This means that the user will have to determine the maximum number of years his earliest year field will antedate his CURRENT-YEAR, and the number of years (max) his last field will postdate it. Say it's five in both cases; he codes:

END CENTURY AT CURRENT-YEAR + 5 WHEN CURRENT-YEAR IS 95 THRU 04

The effect of this clause is to activate the left-rotation of years only in the years xx95 thru xx04. Thus 90% of the time the inefficiency spoken of will be avoided.

Let's look at a hypothetical case. The year is 2000. The fields are DATE-ORDERED and WILL-BE-AVAILABLE-ON. It is a safe assumption that there are no open orders more than five years old, and that there are no out-of-stock situations that will persist for more than five years. The worst case would have a DATE-ORDERED of 95 and a WILL-BE-AVAILABLE-ON of 05. When these are left-rotated by 6 (2005-1999) they become 89 and 99. If the year is 1995, left-rotation of years 90 & 00 by 01 will convert them to 89 & 99, so they will compare as desired. If the year is 2004, left rotation of years 99 and 09 by 10 results in 89 & 99. Since this is the worst case, all other cases are safe.

Probably most cases would have a date-range of less than ten years, so that the length of time that inefficiency will have to be endured will be limited. For situations where the range of dates is longer, it might be worth expanding the fields to four digits. (Hmmm--maybe a four-digit year field in DATE will be needed too?) But shops should have the option of avoiding this disruptive file conversion if they desire.

It would be desirable to leave END CENTURY AT statements in place in programs to simplify things when 2100 rolls around. It would also be desirable to start putting such statements into programs well in advance of 2000, to avert a crisis-fix situation down the road. However, it should be possible to avoid the inefficiency that will result from every reference to a date field causing a check of CURRENT-YEAR. This isn't a serious inefficiency (unlike the one referred to above), but it seems "dumb" & offensive. Users who want to should be able to code a compiler-directing statement to ignore END CENTURY clauses when compiling UNLESS DATE > "900000" (for example). Such users would be trusting themselves to recompile frequently enough to "enable" CURRENT-YEAR checking in advance of its being needed, or they would be intending to keep records of the date-field characteristics of their programs with the intention of recompiling when needed, or both.

I'm certain the ideas here can be vastly improved on. I just want to start the ball rolling in hopes that someone else will run with it.

One concern I have is how to get non-elementary date fields to inherit year-rotation from their year fields, without having inappropriate group items inherit this characteristic too, as when the group item containing a year field is not a date field.

Another concern is how this will fit in with the assembly-language date routines that some shops use for efficiency.

The latter concern would be reduced if COBOL had intrinsic date-handling functions to take over these tasks. Users who shifted over to using these functions would be working with a known quantity, which could be interfaced properly with the END CENTURY clause by the Committee.

Date-handling functions were suggested in (83030) Suggestion Smorgasbord, p. 2, #9. The advantages of using functions for this task were not fully enumerated there. They include:

- Greater program portability (The program is more self-sufficient.)

- Greater programmer portability (Less shop-specific stuff to learn.)

- Greater efficiency for most shops

- Less opportunity for computer crime. (Monkeying with an assembly-language date routine, or invoking a special copy of such a routine, are hard-to-detect ways a criminal might increase interest payments into his account, etc. Since such routines aren't standardized, detection would be harder for auditors. If the routine is in the language, monkeying will be harder & detection easier, I'd think.)

- More features (The CC can do a fuller job than most shops.)

These advantages would apply to State-name functions too, mostly.

Another concern is reruns; CURRENT-YEAR might have to be adjusted to reflect some time in the past. Testing would necessitate adjustment of CURRENT-YEAR into the future.

CODASYL

P.O. BOX 1808

WASHINGTON, DC 20013

November 18, 1983

COMMITTEE COBOL

REPLY TO

General Services Administration
ATTN: OIRM/KMPS
18th and F Streets, NW
Washington, DC 20405

SUBJECT: Minutes of Meeting
November 14-17, 1983
Monterey, California

Chairman: D. F. Nelson, Tandem Computers, Inc.
Vice-Chairman: W. C. Rinehuls, General Services Administration
Secretary: W. C. Rinehuls
Asst Secretary: V. Gwillim, National Computing Centre
Archivist: C. A. Calder, Norfolk Southern

MEMBERS ATTENDING

Burroughs Corporation (BUR)
Canadian Federal Government (CFG)
Control Data Corporation (CDC)
Digital Equipment Corporation (DEC)
General Services Administration (GSA)
Honeywell Information Systems (HON)
International Business Machines (IBM)

Micro Focus, Ltd. (MF)
National Bureau of Standards (NBS)
National Computing Centre (NCC)
NCR Corporation (NCR)
Norfolk Southern (NS)
Sperry Corporation (SY)
Tandem Computers, Inc. (TAN)
United States Navy (USN)
Wang Laboratories (WANG)

	<u>11/14</u>	<u>11/15</u>	<u>11/16</u>	<u>11/17</u>
A. Rockall	A	A	A	A
N. Powroz	M	M	M	M
D. Walls	M	M	M	M
C. Rice	A	A	A	A
W. Rinehuls	M	M	M	M
P. Hall	M	M	M	M
M. Sander	M	M	M	M
A. Reimann	A	-	-	-
A. Fryer	M	M	M	M
M. Vickers	M	M	M	-
V. Gwillim	M	M	M	M
R. Paul	M	M	M	M
R. Kurz	A	A	A	A
J. Brieschke	M	M	M	M
D. Nelson	M	M	M	M
J. Greco	A	A	A	A
C. McComas	M	M	M	M

M = Member
A = Alternate
- = Absent

MEMBERS NOT ATTENDING

Deloitte Haskins and Sells (DHS)
Four-Phase Systems (FPS)
Jerome Garfunkel Associates (JGA)
United States Army (USA)
United States Air Force (USAF)

LAST DATE ATTENDED

September 1983
May 1983
September 1983
September 1983
September 1983

Minutes of Meeting
November 1983

VISITORS ATTENDING

M. Holt
R. Knights
B. Laschkewitsch

REPRESENTING

Data General Corporation
R. Knights
Tolerant Systems

DATE

Nov 14-17
Nov 15-17
Nov 14

I. REVIEW AND APPROVAL OF AGENDA

The following agenda was approved:

I. Review and Approval of Agenda

II. Review and Approval of Meeting Minutes

III. Administrative Reports

A. Chairman

1. (83146) CC-83002 - Scope and Program of Work 1984
2. (83147) CC-83003 - Annual Report

B. Secretary

C. Liaison

1. ANSI X3
2. ANSI X3J4
3. ECMA TC 6
4. British Computer Society

D. Task Groups

1. Screen Management

E. Meetings

F. Publications

IV. Old Business

A. Priority Items

1. (81039) LEI-81001 - STOP RUN WITH ERROR STATUS

B. Bylaw Changes

1. (83108) GSA-83005 - Bylaw Change - Officers
2. (83119) RK-83014 - Editorial Guidelines
3. (83143) CFG-83003 - Bylaw Change - Editorial Guidelines
Page Numbering

C. ANSI Priority Items

1. (83144) ANSI-83010 - USE Procedures and External Files
2. (83145) ANSI-83011 - PICTURE Character P
3. (83140) TAN-83027 - RECORD Clause - One Step Back
4. (83138) TAN-83025 - SEARCH Item With > 1 Subscript
5. (83139) TAN-83026 - Comparison in SEARCH Statement
6. (83061) TAN-83009 - HIGH-VALUES in SPECIAL-NAMES
7. (83018) TAN-83003 - Variable-Occurrence Evaluation
8. (83062) TAN-83010 - VALUE and OCCURS DEPENDING ON
9. (83060) TAN-83008 - DIVIDE Remainder

5. (83118) RK-WP - Turn-of-the-Century Problem

CCC MINUTES FOR

83-11,

#36

After discussion, the Committee considered action on this Working Paper completed.

(= REJECTED)

COBOL COMMITTEE
WORKING PAPER

ITEM #: 84051
AUTHOR: Roger Knights
SUBJECT: Date-Handling Functions

ORG #: RK-WP
DATE: Oct. 2, 1984

A family of date-handling functions was authorized at the 83-11 meeting (p. 35 of minutes). I hope this paper will get the ball rolling, even though its direction may well need considerable correction. I suggest handling three main date formats.

1. YR-DAY and YEAR-DAY are inferred from lengths of 5 and 7 (using PIC 9 (USAGE DISPLAY) or X). [2- & 4-digit years respectively]. (The leftmost 2 digits of year, if omitted in an argument, are assumed to equal those of the century in CURRENT-DATE.)
2. YR-MO-DA and YEAR-MO-DA are inferred similarly from lengths of 6 and 8. "Year first" is the standard internal Cobol date sequence, and is also the ISO standard all-numeric date format. (Hyphens are the official ISO separators.)

If a 6- or 8-digit date argument is in "customary" American or European format, a parenthetical suffix to the argument may be used, e.g. "(MDY)" or "(DMY)". (Internally these would simply resequence the fields of the argument before invoking the corresponding year-first function.) For symmetry the default suffix "(YMD)" may also be used.

If it is desired to have the value returned be in "customary" format, the four functions DA-MO-YR/YEAR and MO-DA-YR/YEAR may be used. (Internally they would simply invoke YR/YEAR-MO-DA and resequence its output, just as the "YR" functions simply invoke the "YEAR" functions and chop off their left two digits. These all spare the coder the trouble of having to do it himself.)

3. ABS-DATE is inferred from PIC 9(9). (Or perhaps it could also be inferred from the combination of 9(7) and a USAGE of BINARY or PACKED (to avoid confusion with format 1), since it would rarely be used externally, and since this would allow storage in 3 bytes.)

The absolute date for Jan. 1, 1990 is 2,447,893. Absolute date 0,000,001 was Jan. 1, 4713 B.C. I attach a copy of a letter from Computerworld by Richard L. Conner containing more information on the absolute format. (Conner has given me some tips on this paper.) The absolute format would be a useful base for users of Islamic & other non-Gregorian calendars to work with, as well as being the

FID

most efficient method for Westerners who take the trouble to convert to it, so it should be allowed even though not as common as the other two.

Note that the modulus 7 of any absolute date, plus 1, yields the day-of-the-week number returned by CURRENT-DATE. I've used the term "absolute", which is its name in astronomy, rather than "Julian", to avoid confusion in those shops that use the latter term as a synonym for YR-DAY.

The date-format of the arguments is inferred from their length or data-class, as outlined above. Fortunately, they all differ. As mentioned, the 6- and 8-digit arguments may be qualified by "(DMY)" and "(MDY)".

The first argument of all the 13 functions I suggest may be in any of the three date formats mentioned above. This symmetry should make things easy to remember. (Although it's not necessary to invoke a function to do date math on absolute dates, which can be added and subtracted directly, I suggest allowing such absolute arguments for symmetry.)

The argument(s) and the format returned by the first 9 functions are indicated below. The functions' names indicate the values returned.

Names	Arg-1	Arg-2	Arg-3	Returns
YR-DAY	Any of the three date formats.	"PLUS" or "MINUS".	9999 thru 9 days (optionally signed).	9(5) [9(7) = YEAR-DAY]
*YR-MO-DA			Any of the 3 date formats.	9(6) [9(8) = YEAR-MO-DA]
ABS-DATE		"TO" X	Any of the 3 date formats.	9(9)
DAYS-FROM			Any of the 3 date formats.	S9(4)
IF-DATE		<rel. oper.>		true or false

*[Or MO-DA-YR/YEAR or DA-MO-YR/YEAR.]

Here are some examples. If only one argument is specified (as in the first six functions below), only a format conversion without addition or subtraction is performed. The format of arguments is spelt out beneath or alongside them, in brackets.

Function	Args.	Returns
1. YR-DAY	(840201) [YR-MO-DA]	84032
2. YR-MO-DA	(84032) [YR-DAY]	840201
3. YR-DAY	(000000001) [ABS]	9999999
[I suggest such BC dates return 0 or all 9s--else dates will have to return a leading space or a minus.]		

Function	Args.	Returns
4. ABS-DATE	(19830101) [YEAR-MO-DA]	002445336
5. ABS-DATE	(83001) [YR-DAY]	002445336
6. ABS-DATE	(2445336) [ABS] [Should this sort of thing be forbidden? --Args. of the same format as the result, I mean.]	002445336

Here are the 1st 3 functions above, with 3 arguments;
DAYS-FROM and IF-DATE always take 3 arguments

7. YEAR-DAY	(83362 PLUS 5) [YR-DAY]	1984002
8. YEAR-DAY	(83001 PLUS 365.2524 * 5) [YR-DAY] [Expressions are OK in a numeric argument--useful. To avoid any ambiguity, PLUS & MINUS in arg-2 are spelt out.]	1988001
9. YR-MO-DA	(840101 MINUS 2) [YR-MO-DA]	831230
10. YR-DAY	(840105 PLUS 395) [YR-MO-DA] [Extra day in leap year affected result.]	85034
11. DAYS-FROM	(850203 TO 840105) [Both in YR-MO-DA] [Negative value returned since 2nd arg. is lower; Both args. must have same format else 9999 returned.]	-395
12. IF-DATE	(84001 = 010184 (DMY)) [YR-DAY] [DA-MO-YR]	true

A fourth argument might be added (eventually, if necessary) to the functions above: VTOC IS <00-98>, where VTOC stands for "Virtual Turn of the Century". This would cause all dates with years below the VTOC year to be considered to be higher than the year ending in 99. A VTOC of 00 could be coded as a placeholder to indicate that it should be modified at a future date, but it would be ignored by the compiler. By setting the VTOC to some year ahead of all dates in the input, the turn of the century could be "gotten around". Arithmetically lower but temporally higher 2-digit years in the new century would be treated as higher by the function. After all dates in the input had moved into the new century (so that arithmetic comparisons would return the desired results again) the VTOC operand could be reset to 00.

A book published a few months ago (COMPUTERS IN CRISIS BY J. & M. MURRAY, WHICH I'LL BRING TO THE MEETING - ATTACHED IS REVIEW IN CW) is devoted entirely to the mess the turn of the century is going to cause. It estimates the conversion cost as being in the billions. If this VTOC argument could save a small percentage of that, it would be worth it. This is a much more modest proposal than the method I suggested last year, which would have retrofitted existing code with a new clause.

The last four functions, below, are the simplest. They can take only one argument, and what they return is obvious from their names.

Names	Arg-1	Returns
DAYS-IN-MONTH	<div style="display: flex; align-items: center;"> <div style="border: 1px solid black; padding: 2px; margin-right: 5px;">Any of the 3 date formats</div> <div style="margin: 0 10px;">OR:</div> <div style="border: 1px solid black; padding: 2px;">99 (1-12) 9 (1-7) 99 (1-12)</div> </div>	99 (1-31)
DAY-NO		9 (1-7)
DAY-NAME		A(9)
MONTH-NAME		A(9)

The coder can truncate DAY-NAME & MONTH-NAME's returned values to the length of his target item. If he wants unequal-length abbreviations he can do it himself, later. (Or he can propose a fancier function to the CC. Or resubmit John Piggott's Picture symbol L, which truncates trailing spaces within a picture, so that the day of a month would always follow the month name with just one space.)

The coder could achieve the effect of these two NAME functions by defining and searching a table himself, after converting the dates (if necessary) with one of the first three functions. But I think the directness of the NAME functions is useful and stylistically OK. If most common tasks could be identified and functional-ized, the language would be higher level and coding would be largely a matter of slapping functional "boilerplate" together, rather than traditional "programming". This is the "reusable code" idea, which seems to be a coming thing.

Even though there is some overlap in all these functions, it is better to have that and let the functions internally invoke one another (when necessary) than to squeeze such redundancy out and force the coder to nest his functions two or three deep, which defeats the purpose of the higher level approach that functions give. (An example of such invocation is that the two MONTH functions above would have to invoke YR-MO-DA if their argument were a date in YR-DAY or absolute format, and the two DAY functions would have to invoke ABS-DATE if not passed an absolute date argument.)



OFF THE PRESS
George Harrar

BOOK REVIEWS

COMPUTERS IN CRISIS

By Jerome T. Murray
and Marilyn J. Murray

The subtitle — "How to Avert the Coming Worldwide Computer Systems Collapse" — teases the reader with the threat of unnamed calamity. As in all good doom-predicting books, there is, of course, the promise of a solution.

The crisis will be triggered on Jan. 1 of the year 2000 by a simple situation: Programs rely on six-digit date fields. Which date is later, 12/31/99 or 01/01/00? Programs would say the first; programmers and everyone else naturally answer the second.

Sixteen years seems sufficient time to solve this problem. But the Murrys foresee trouble: "The true potential for disaster lies in our tendency to procrastinate... our readiness to 'patch' or treat symptoms until it is often too late to [eradicate] successfully the disease."

The disease's course is poetically described: "We may well expect widespread suspension of computer processing in the year 2000 and beyond, with many terminal screens as dark as a villain's heart."

The six-digit dilemma affects almost all industries by disrupting the aging of accounts receivable and payable, pensions, benefits eligibility and so on. It is a user problem, but one that could use vendor help.

The authors write "An Open Letter to Our Leading Vendor" in which they ask, among other changes, "Please replace the current system-generation option for the six-digit IPL date: mm/dd/yy or dd/mm/yy or yy/mm/dd, with the eight-digit option, mmdyyy or dmmyyy or yyyymmdd."

Even if vendors come to the aid of their users, the conversion still will be a mammoth task of altering data, invading source logic and modifying input protocols. An example is cited of a Fortune 500 company with a library of 50,000 Cobol programs, averaging 750 lines of code each for a total of 37,500,000 total lines. Figuring a productivity rate of 15 debugged lines per day, the authors calculate more than 21 years of work by 500 programmers to rewrite it all.

The book overdramatizes the date-digit problem, in part to draw attention to it. Others have ventured the same thesis more quietly and stirred up little or no interest.

Much of the book can be overlooked. The preface lays out the problem well; Chapter 1 provides interesting background on time and its calculation, and Chapter 11 prescribes a conversion solution for users. Read those sections if nothing else.

Hardcover, 360 pages, \$32.95, ISBN 089433-223-6, Petrocelli Books, Inc., 1101 State Road, Princeton, N.J. 08540.

GETTING STARTED WITH THE IBM PC AND XT

Edited by David Arnold

In partnership with *PC World* magazine, Simon & Schuster, Inc.

launches the PC World Library with this introduction to IBM's Personal Computer and Personal Computer XT models.

This attractively designed first volume is an unabashedly friendly and positive look at what novice users can do with their new machines. This is a hand-holding book that puts pages of definitions where the reader needs them — before the technical sections rather than tucked away in an appendix.

Future volumes in the *PC World*-Simon & Schuster series promise to cover desktop applications, communications and hardware for the Personal Computer and XT. Additional books will feature PCjr — how to get started and have fun with it.

Paperback, 183 pages, \$14.95, ISBN 0-671-49277-2, Computer Book

Division, Simon & Schuster, 1230 Ave. of the Americas, New York, N.Y. 10020.

AN INFORMATION SYSTEMS MANIFESTO

By James Martin

There is little reason to paraphrase James Martin. He is that rare writer in the computer field, one who expresses himself clearly and forcefully. He writes:

"What is needed instead is freedom for user areas to employ their own initiative in creating the systems they need, but they still must obey certain laws. They make systems conform to a set of rules that permit them to interchange data, now and in the future.

"Perhaps the most notorious class

of systems that don't work is management information systems.

"It is the job of management to build a computerized corporation, and the foundation stone of that is the data models that are used.

"Prototyping is by far the most effective for giving the user a clear, realistic representation of the system that is to be built.

These quotes reflect this book's
See **BOOKS** page 44

Publishers wishing to have their books considered for review can direct books, prepublication galley, press releases, catalogs or other information to George Harrar, Book Review Editor, Computerworld, P.O. Box 880, 375 Cochituate Road, Framingham, Mass. 01701.

Getting timely information to 50,000 people who need it is an event in itself.



Behind the action, pageantry and spectacle of the 1984 Olympic Games is one of the most extensive data communications systems ever built.

The Electronic Messaging System—12 computers, 1,700 terminals, 300 printers—was created for the 1984 Olympics by AT&T. And tied together with Infotron networking equipment.

The EMS is designed to replace the old system of hand-carried reports in the world's first multi-site Olympics. It does everything from displaying event results, to supplying athlete biographies for reporters,

to providing schedules, qualifying information and personal messages to the participants themselves.

The system's true complexity lies in the numbers of people it serves: the 50,000 officials and reporters, coaches and athletes of the Olympic family. But the demands placed on it are very much like the demands all Infotron customers place on their networks.

Performance. Flexibility. Reliability.

Advanced Network Integration from Infotron—a sound basis for networks of any size or configuration. Simple.

Or breathtakingly complex.

We're proud of the part we're playing.

Infotron.
First in performance and reliability.

Infotron Systems Corporation,
9 North Olney Avenue, Cherry Hill, NJ 08003.
Or call toll-free: 1-800-345-4630.

Send for our informative book,
Making It Through the Maze of Data Communications.

Name _____
Title _____
Company _____
Address _____
City _____ State _____ Zip _____
Telephone _____

CW6/11

INFOTRON SYSTEMS

84-5-14

CW

Supplement to (84051) RK-WP Date-Handling Functions

Since submitting the above paper I have received & read Computers in Crisis by J. T. and M. J. Murray. (It can be ordered from the publishers directly--see the price & address in the review attached to my WP.) I attach many relevant pages from this important work. (Over half the book is pseudocode and (IBM) ALC listings of subroutines that sites can call to perform the date-handling functions described by the authors.)

I list below my points of agreement and disagreement with the authors.

1. **DISAGREE:** They suggest calling the YEAR-DAY format "Neojulian". (I think "Long Julian" would be better.) But preferably we should avoid all such idiosyncratic and non-parallel names (like their "commercial", "FIPS", and "European") and use instead names whose length and content contain useful information, such as MO-DA-YR, YR-MO-DA, and DA-MO-YR, and use YEAR-DAY rather than "Neojulian".
2. **AGREE / DISAGREE:** They suggest storing YEAR-DAY in a 4-byte packed field, rather than requiring DISPLAY. They claim that because some sites store 6-digit dates in packed 4-byte format, 4 bytes is the maximum that a replacement format should take. This point may be sound, but since some machines (e.g. micros) lack the ability to work with packed fields, it would seem better to use binary. The righthand two bytes would be the day, and the lefthand pair the 4-digit year.
3. **DISAGREE:** They believe that data in external storage ~~must be converted to carry a 4-digit year~~, to enable valid date handling across the turn of the century. This is the basis for their difficult-to-do conversion plan (aptly described in "Chapter 11"). The simpler method is to convert only programs (to use date functions) and to let those functions contain a Virtual End Of the Century parameter (VEOC). In this way the worst impact of the turn of the century problem may be "dodged". The "mapping" (done within the function) of a virtual century (whose first year, say, is 03 & whose End year is 02) to the digits 00-99 can be accomplished in a half-dozen instructions or less. (Subtract the End year value + 1 from all years & add 100 to negative results.) (I now believe that specifying the end-year of a century is clearer than the method of my WP, which suggested specifying the Turn or start-year (00).)

One difficulty with converting files is that all programs that use those files must be identified, frozen, and switched over first. With the date-function dodge, they can be

converted one at a time. R. L. Conner agrees with me that this is a more practical solution for the real world (of conversion—allergic managers) than the method he personally prefers (converting files to an absolute date format). The penalty of "dodging" is merely the processing time taken in mapping virtually high years like 03 to really high years like 99. This is minor compared to a file conversion effort. And the processing penalty is temporary--once the turn of the century is out of the way, sites can drop the VEOC parameter and continue on their merry way. (Sites that want to convert to a more sensible date format can still do so, of course, and make use of the functions I've suggested as well.)

4. **AGREE:** They suggest starting what I call the "absolute" date at Jan. 1, 1601, rather than at the astronomer's traditional Jan. 1, 4713 B.C. This may simplify and speed some computations, be mentally simpler for coders to grasp, and will enable the date to be stored (as BINARY) in 3 bytes rather than 4. This will allow 2700 years or so before the absolute date wraps around to zero. (Probably 4 bytes should be what the function expects, though.) The division of such dates by 7 still yields modulus values that match Cobol's day-of-week conventions (1=Monday, etc.). (See p. 130 of the book.)
5. **DISAGREE:** They suggest an upper limit on YEAR of 3400 AD, since 3400 should not be a leap year (it will get us 1 day out of synch with the sun) although the current formula says that it should be. I assume the ISO will have issued a standard correcting the matter by then. (Maybe they could be requested to do so now, to enable stable universal date routines to be written. Meanwhile, it could just be assumed that they will do so.)
6. **AGREE:** They provide return codes for all their "functions"; these give different values for all possible errors in the input. In some cases there are nearly 10 such codes. It seems to me that the CC will ultimately have functions where there are also numerous error possibilities, and where input could not be checked without nearly duplicating the action of the function itself. (I.e., VALIDATE is insufficient.) There should be special registers associated with such functions whose names are made up of the name of the function plus some unusual prefix or suffix. For consistency, current functions should perhaps also switch to this method, rather than returning a special error code as their function-value.

Perhaps there could be a single return code special register, the meaning of whose values would depend on the function last referenced. Since nested functions might be executed, the first part of the special register would be a code indicating the function that last placed a value in it.

7. **AGREE:** They suggest a function to return the difference between two dates in terms of the elapsed years, months, and days between them, as well as one that returns the number of days. This is useful for "long interval aging". (I attach the relevant pages from the book.) (79-85 & 90.)

I considered suggesting a function to do this, but was defeated by the anomalies inherent in the calculation, which made me unsure of how to define the result. The first two functions on p. 2 of my WP (YR-DAY & YR-MO-DA and their variants) should simply allow "any date format" in argument-3; they would then return date differences (or sums) in YR-DAY and YR-MO-DA format (and their variants).

8. **AGREE:** They provide a function that returns true if a date is a holiday or not. The user must fill in a table parameter indicating which of many possible holidays in the US, Canada, etc. he wants to the function to check for. (The model subroutine they provide includes a sophisticated calculation for the date of Easter.)

I omitted this function from my list because its table argument requires some unusually "peculiar" information that I didn't possess, and because I felt CC members might object to it because some fuzzy judgment will be required in determining its allowable candidate holidays. (The "where do we stop?" concern.) Overweighing that is the desirability of even an imperfect function of this kind. As the authors write:

"As we relinquish more and more control of our destiny to computers, we must humanize them. ... It is time now for the development of some user sympathetic software. Recognition of our holidays and celebrations is but an elementary step in this direction. Like so many broadly accepted notions, this too has an economic basis."

I suggest the function:

IF-HOLIDAY (argument-1 argument-2)

Argument-1 (as always) is any of the three date formats. Argument-2 is the name of a table containing coded entries representing (according to some convention yet to be determined) the holidays that are to be considered as candidates by the function.

I've spoken to J.T. Murray, and he has indicated his willingness to attend a CC meeting to discuss the seriousness of the problem and his book. (He is Director of MIS at The Rose Packing Co. / 453 Raintree Dr. / Glen Ellyn IL 60137 / 312-381-5700.) The publisher has given me permission to have the attached xeroxes distributed to the CC, and is willing to sell 20-odd copies of the book at wholesale to the CC. (THEY would be sent to a meeting site.)

Advantages of Julian Dates

We should use Julian dates instead of the Gregorian calendar in our data bases and programs. I'll explain why, but first I'll clear up a possible confusion regarding the term "Julian date." Many people in our business believe that a Julian date is composed of the year and the day of the year; for them the "Julian date" corresponding to Jan. 1, 1983 (Gregorian) is [19] 83.001.

That is not a Julian date. A Julian date has no year in it; it is a serial number associated with a given Gregorian date, serial number 1 having been assigned to a day thousands of years in the past. The Julian date for Jan. 1, 1983 is in fact 2 445 336.

Now for some advantages of using Julian dates. I emphasize that I do not advocate entry or display of Julian dates. I merely urge that all dates

be carried internally in data bases and so on in Julian form and converted from and to Gregorian form as necessary.

Many applications require the number of days between two dates or the date some number of days before or after a given date. Most shops have more or less elaborate subroutines to do the calculations. Using Julian dates, the process is simplicity itself — just add or subtract.

It's often necessary to determine the day of the week on which a given date falls. The information is much more easily obtained from the Julian form than from the Gregorian form. Just take Julian date modulo 7. If the result is zero, the day of the week is

Monday; if 1, it's Tuesday.

For binary hardware, the Julian date will fit nicely into 32 bits. Indeed, 24 bits will do through Monday, May 9, 4122.

The 21st century looms. Some problems may arise from the widespread practice of carrying only the last two digits of the year. These problems wouldn't arise through the use of Julian dates.

Interconversion of Gregorian and Julian dates is easy. Henry F. Fliegel and Thomas C. Van Flinders published a pair of excellent algorithms in *IBM Comm ACM* 657 (October 1968). As an exercise in hand optimization, I coded the algorithms as entrant subroutines for IBM 360/370

LETTERS

CPUs, invocable by Cobol, Fortran, PL/I or assembler callers.

Together, the subroutines occupy about 320 bytes. On an IBM 3033 CPU, Gregorian to Julian conversion takes about 12 microseconds; going the other way takes about 14.

The Fliegel and Van Flinders algorithms are nicely tolerant. For example, if a bank lends on a 180-day note on Nov. 22, 1982, it can find the due date by feeding Nov. 202, 1982 to the Gregorian-Julian algorithm, which will absorb it without a hiccup.

Data base designers might do well to carry Julian dates in the physical data base, accepting and presenting Gregorian dates in the logical counterpart, thus rendering interconversions invisible to the user.

Richard L. Conner
San Francisco, Calif.

(I forgot to attach this to my WP.)

FINAL POSTSCRIPT:

AN ATTRACTIVE (?) ALTERNATIVE TO THE USE OF KEYWORDS IN DATE FUNCTIONS TO INDICATE FORMATS WHERE LENGTH IS THE SAME (YR-MO-DA, MO-DA-YR, DA-MO-YR, etc.) WOULD BE TO ADD A CLAUSE (OR MODIFY PICTURE) TO PROVIDE TO FORMAT INFORMATION WITH THE ITEM ITSELF, E.G., IN PICTURE THE FOLLOWING CONVENTIONS COULD BE USED:

ABSOLUTE DATE
DAY OF YEAR (1-366)
DAY OF MONTH (1-31)
DAY OF WEEK (1-7)
MONTH (1-12)
YR (00-99)
YEAR (1601-9999)
1 2 3 4 5 6 7 8

THESE SYMBOLS WOULD BE TREATED AS THOUGH THEY WERE 94, OUTSIDE OF DATE FUNCTIONS (& POSSIBLY VALIDATE). DATE DATA TYPES ARE COMMON IN NONPROCEDURAL & MICRO USER TOOLS, SO THIS

IS NOT A RATIONAL TERMINATION

C O D A S J L

P. O. BOX 3609
NORFOLK, VA 23514-3609

November 16, 1984

COMMITTEE
COBOL

REPLY TO
General Services Administration
ATTN: OIRM/KMPS
18th and F Streets, NW
Washington, DC 20405

SUBJECT: Minutes of Meeting
November 12-15, 1984
San Jose, California

Chairman: D. F. Nelson, Tandem Computers, Inc.
Vice-Chairman: W. C. Rinehuls, General Services Administration
Secretary: W. C. Rinehuls
Asst Secretary: V. Gwillim, National Computing Centre
Archivist: C. A. Calder, Norfolk Southern

MEMBERS ATTENDING

		<u>11/12</u>	<u>11/13</u>	<u>11/14</u>	<u>11/15</u>
Control Data Corporation (CDC)	D. Walls	M	M	M	M
Data General Corporation (DGC)	C. Symons	M	M	M	M
Deloitte Haskins and Sells (DHS)	R. Tatlow	M	M	M	M
Digital Equipment Corporation (DEC)	C. Rice	M	M	M	M
General Services Administration (GSA)	W. Rinehuls	M	M	M	M
Honeywell Information Systems (HON)	P. Hall	M	M	M	M
International Business Machines (IBM)	M. Sander	M	M	M	M
Micro Focus, Ltd (MF)	A. Fryer	M	M	M	M
National Bureau of Standards (NBS)	M. Vickers	M	M	M	M
National Computing Centre (NCC)	V. Gwillim	M	M	M	M
NCR Corporation (NCR)	R. Paul	M	M	M	M
Norfolk Southern (NS)	B. Moore	A	A	A	A
Sperry Corporation (SY)	J. Brieschke	M	M	M	M
Tandem Computers, Inc. (TAN)	D. Nelson	M	M	M	M
United States Army (USA)	J. Gay	M	M	M	M
United States Air Force (USAF)	M. Lee	M	M	M	M
United States Navy (USN)	J. Greco	A	A	A	-
Wang Laboratories (WANG)	C. McComas	M	M	M	M

M = Member
A = Alternate
- = Absent

MEMBERS NOT ATTENDING

Burroughs Corporation (BUR)
Canadian Federal Government (CFG)
Jerome Garfunkel Associates (JGA)

LAST DATE ATTENDED

September 1984
May 1984
September 1984

Minutes of Meeting
November 1984

VISITORS ATTENDING

R. Knights
K. Lund

S&PC
IBM

Nov 12-15
Nov 12

I. REVIEW AND APPROVAL OF AGENDA

The following agenda was approved:

- I. Review and Approval of Agenda
- II. Review and Approval of Meeting Minutes
- III. Administrative Reports
 - A. Chairman
 1. Scope and Program of Work - 1985
 - B. Secretary
 - C. Liaison
 1. ANSI X3
 2. ANSI X3J4
 3. ECMA TC 6
 4. British Standards Institute
 - D. Task Groups
 1. Screen Management
 - E. Meetings
 - F. Publications
 1. (84056) GSA-84006.00 - Print Errors
- IV. Old Business
 - A. Priority Items
 1. (84008) JSW-8401P - READ PRIOR
 - B. Special Items
 1. (83032) RK-82003 - GROUP INDICATE ON
 2. (84041) RK-WP - EVALUATE: 6 Enhancements
 3. (84042) RK-WP - Wildcard REPLACING In COPY
 4. (84051) RK-WP - Date-Handling Functions
 5. (84057) RK-WP - Suggestion Salad Bar
 6. (83151) RK-83031 - Miscellaneous Editorial Misc.
 7. (84034) RK-84005 - Missed Editorial Misc.
 8. (84038) RK-84006 - EVALUATE: Editorial
 9. (83045) RK-83002 - Add PL/I's REPEAT Function
 10. (84006) RK-84001 - A BLANK/VALID Clause
 11. (84043) RK-84008 - PERFORM: 5 Enhancements
 12. (84044) RK-84009 - SUM Function
 13. (84045) RK-84010 - Boolean Equivalency Operator B-LIKE
 14. (84052) RK-84011 - SPELT-OUT INTEGER Function

4. (84051) RK-WP --Date-Handling-Functions--

84-11 MINUTES

During discussion, the following straw-votes were requested:

Manipulation of dates should be accomplished by data types instead of functions.

FOR: 7

AGAINST: 10

Functions should be included for conversion between standard and absolute date form.

FOR: 11

AGAINST: 1

A function should be included for conversion between standard and Julian date form.

FOR: 11

AGAINST: 3

A function should be added for "Virtual Turn of the Century".

FOR: 1

AGAINST: 14

The Committee considers action on this Working Paper completed.

J4 Response to U.S. Public Review Comments on COBOL WD 1.4

* indicates a change was made in response to an item (Sometimes the * is before the comment number and sometimes it follows the number.)

These public review comments are included in this document, ordered by X3J4 document number:

- 96-0384 Comments on ISO/IEC CD 1989, Prog lang COBOL (Silletti/Wallace)
- 96-0389 Cmts on CD 1989, Prog lang COBOL; IBM Part 2 (Silletti/Wallace)
- 96-0412 - public review comments from David DeJongh
- 96-0414 - public review comments from Raymond Obin
- 96-0428 - public review comments from Lee Hansen
- 96-0429 - public review comments from Charles Townsend
- considered 96-0442 and 96-0441 as part of this comment
- 96-0430 - public review comments from Ronald Silletti - Part 3
- 96-0446 - public review comments from Ronald Silletti - Part 4
- 96-0447 - public review comments from Robert Sandler
- 96-0458 - public review comments from John Piggott
- 96-0461 - public review comments from Jeffrey Friedman
- 96-0473 - public review comments from Don Schricker
- 96-0474 - OO public review comments from Don Schricker
- 96-0476 - public review comments from Ronald Silletti - Part 13
- 96-0477 - public review comments from Ronald Silletti - Part 14
- 96-0478 - public review comments from Ronald Silletti - Part 14
- 96-0479 - public review comments from Roderic Knights
- 96-0480 - public review comments from Ronald Silletti - Part 11
- 96-0481 - public review comments from Ronald Silletti - Part 10
- 96-0483 - public review comments from Ronald Silletti - Part 5
- 96-0484 - public review comments from Ronald Silletti - Part 9
- 96-0485 - public review comments from Ronald Silletti - Part 7
- 96-0486 - public review comments from Ronald Silletti - Part 8
- 96-0487 - public review comments from Wataru Takagi
- 96-0488 - public review comments from Ronald Silletti - Part 6
- 96-0489 - public review comments from Ronald Silletti - Part 12
- 96-0490 - public review comments from Clark Morris
- 96-0493 - public review comments from Ronald Silletti - Part 15
- 96-0500 - Public review comments from Micro Focus (Gilbert/Gamble)
- 96-0502 - Public review comments from Ronald Silletti - Part 16
- 96-0506 - Public review comments from Jonathan Beit-Aharon

sent Nov. 20, '96

96-0384 R. F. Silletti (IBM Corporation)

*1. Length of figurative constant. The draft does not specify the length of a figurative constant when used in a concatenation expression. IBM has submitted proposal X3J4/96-0342 to Technical Committee X3J4 to amend the document.

Response: Accept. The length of a figurative constant in a concatenation expression was specified as one in 97-0218, Length of figurative constants in concatenation expressions, which was approved by J4 at meeting 208.

November 5, 1997

Document: J4/97-0
page 46 of 10

- Add four methods (I can't think of any pleasing names) similar to the methods of Comparable, all invariant, but which ignore upper/lower case for the purposes of the comparison.
3. I'm a bit disturbed by the absence of any streaming methods on an object for the purposes of externalization and subsequent reconstitution. These protocols might be used on transient (non-named) objects, not just formally persistent objects.)
4. P 587, 16.3.1.1. Why is "OccurrencesOf" not invariant? How does this method modify the collection self? Same comment for List (p 633, 16.3.16.1).
5. P 623, 16.3.14.1. Why is "Intersection" structurally different from all the other operations that produce a changed IdentitySet, for example, Union? Is it just so that the postconditions can be stated neatly?!

Suggestion. Change to ...

AnIdentitySet "Intersection" using Another instead of ...

AnIdentitySet "Intersection" invariant using Another returning AnIntersection

6. P 625, 16.3.14.6.1. Change "method-id. is-subset" to "... IsSubset".
7. P 633, 16.3.16.1. Method "Sublist" should be invariant, right? Just like "OccurrencesOf".
8. Both national character handling and internationalization are important aspects that must be addressed. Dependencies, if any, on the environment of the invoking routine need to be specified.

It's not clear how collating sequences are managed, even with respect to non-internationalized environments. Is the collating sequence in the library always the one in effect at the time of invocation, thus requiring the library to be an "internationalized" implementation with respect to collating sequence? How are the different character types handled (alphanumeric and national)?

96-0478 - public review comments from Roger Knights

Here are some comments on your draft Cobol-97 standard. They are in four categories: first, Substantive (mostly date-related); then editorial suggestions dealing with: Miscellaneous, your Substantive-Changes-Not-Affecting list, and RW / Validate.

You needn't write me anything but the briefest note about your action on this letter; you have too much other work to do.

A. Substantive:

1. To facilitate Year-2000 conversion-testing, allow the CURRENT-DATE special register to be both modified and tested-to-see-if-modified. (John Piggott suggested the first of these.) I guess the SET statement could be used for the first purpose. For the second, IF would interrogate a new special register that would be either a True/False "indicator" or a backup location like TRUE-CURRENT-DATE or SAVED-CURRENT-DATE or ACTUAL-CURRENT-DATE. And perhaps there should be some standard way to restore CURRENT-DATE to its original status.

Response: Reject.

2. Your FORMAT clause will assist Year-2000 migration by removing an obstacle to the use of compact date forms, such as the Gregorian.INTEGER format used by COBOL's Functions, and the new Packed and Binary formats that various Year-2000 vendors and writers are proposing. The obstacle is that such formats are harder to download to the PC, which has caused some authorities to advise against converting to them. (E.g., See Tick Tick

Tick, Winter '95, page 8, col. 3.) So don't drop it if someone points out any problem areas it may have. Resolve such questions of interpretation in a subsequent Year-2000 task group (see next item).

Response: Accept. The FORMAT clause is retained.

3a. Consider setting up a Task Group to interface with persons and organizations dealing with the Year-2000 Problem, in case there are other changes like the above that could be implemented as Addenda, or at least so that vendor extensions could be coordinated and informally standardized. Most of the work could/should be done on the Internet, with a special extra site for private (inter-member) task group communications. At a minimum, such a task group could consider minor twiddling like the CURRENT-DATE adjustments I proposed in Item 1, above. More substantial enhancements might well be considered, either as a result of user pressure and/or of the opportunity to make a buck by offering a desperately needed service. It may be that these enhancements will come too late to help most users. Still, they will help some, and they will enable those applications that have been deactivated as a result of "triage" to be reactivated, since less work and (especially) no file conversion will be needed to get them running. The enhancements I have in mind include:

Response: Reject. There are already a number of task groups in the industry.

3b. An extensive family of built-in date manipulation functions. In (84051) RK-WP Date-Handling Functions, I suggested (in addition to suggesting the current date-conversion functions) a day-counting (between dates) function and a date-comparison function, both of which could accept operands in different formats. Those two basic items could be added in an addenda, I trust. Beyond them are more sophisticated routines that really ought to be made available—there's certainly a demand for them. There are now several vendors offering date-routine collections, including holiday tables that work both domestically and/or internationally. (User adjustments to them are possible too.) Perhaps one of these vendors would agree to license his product at a low rate to those implementors who prefer to buy rather than build. (Such inexpensive licenseability is commonly considered when ANSI hardware standards are developed, so I extrapolate that it's OK for software too, at least in an emergency. Alternatively, end-user sites could be asked to contribute their date routines to the TG. E.g., as I wrote in (83030) RK-WP Suggestion Smorgasbord, Item 9: "I believe Boeing has a collection which will handle everything but Mayan.")

The advantages of making these intrinsic functions are efficiency, reliability, simplicity (fewer CALLs and COPYs), inter-shop standardization (hence reduced training costs and merger-time nightmares), etc. As a business-oriented language, inclusion of such business-oriented routines is acceptable—indeed, desirable. (Other business-savvy languages have such a data-type; e.g., Clarion, as I mentioned 'way back when.) Year-2000 conversion software could generate these functions and thereby upgrade users' software libraries in the process. Currently, code often comes back from such a conversion in a downgraded (patched-looking) state. In the Nov. '84 minutes, p. 16, the CCC accepted only a subset of the date manipulation functions I had suggested; the other suggestions were implicitly rejected by consensus and not brought to a vote.

Response: These will be considered as an enhancement for future standard.

3c. A DATE data-attribute (i.e., a new clause) whereby a user can both specify an item as being a date and also indicate its format. This would be helpful to the date-comparison and day-counting functions suggested above; although in the absence of a date attribute, the functions could infer one from the item's PIC & USAGE. It would also assist other date-related functions that might be added, as well as being helpful to the maintenance programmer and to COBOL-analysis software. Also, Year-2000 conversion software could insert this data-attribute on items it decides (in optional consultation with humans) are dates.

In addition, the attribute could (if extensible) integrate the many idiosyncratic date formats that now exist into standard COBOL in a nearly seamless way. E.g., assume the keyword for the attribute is "DATE-<suffix>", where standard terms for the most common "suffixes" (like JULIAN and YRMODA and YEARMODA) are built-in, and where other suffixes could be added on the fly. (Perhaps an interface could be provided that would allow user-written (or ISV-written) routines to decode/encode other formats to dates with particular user-suffixes.)

The CCC voted down (10-7) using a data-attribute to handle the four date conversions it now provides in favor of functions in the Nov. '84 minutes, p.16.

Response: A DATE datatype will be considered as an enhancement for future standard. In the meantime, objects, user-defined functions, and TYPEDEFS allow for user-defined date datatype. Consider the SQL date datatype and the full set of functions to access it.

3d A compiler directive (and/or a run-time ability) to set a "sliding window" for two-digit years. The compiler would then be able to adjust arithmetic operations and comparisons and sorts on fields that use such an item to give the desired results. (In those (rare) cases where it can't figure out what's going on it could give up and let the user know he needs to clarify matters.) This is important; it is now too late for much of the user community to make a gradual migration and associated file-expansion to a four-digit year. File expansion terribly complicates the Year-2000 process; it expands the impact onto other programs that accept the files or screens produced, it impacts report formats, and (therefore) it expands the scope of what must be tested. Testing, including regression testing, is estimated to be the hardest part of the Year-2000 process. Imagine how hard it will be to create an entire new environment with a multitude of expanded files, databases, screens, reports, JCL, etc., and then to get the machine time to run realistic simulations. Many applications will simply be abandoned unless there is some way they can "work around" the year 2000 without file expansion.

I suggested this be done in (83118) RK-WP Turn-of-the-Century Problem. It received the "After discussion ... completed" treatment (= consensus rejection) in the Nov. '83 minutes, p. 36. I repeated my suggestion, but only for a function, in 84051. It was rejected 14-1; see Nov. '84 minutes, p. 16. At that meeting I distributed (with authorial permission) substantial photocopied extracts from Jerome Murray's Year-2000 alarm-book, Computers in Crisis. I quoted his worldwide-conversion-cost estimate of \$60 billion and got a roomful of "get-outta-here" looks. I hope, while there is still time for COBOL to be of some help to some users (which means it can save billions at a cost of ten-thousands) that action will be taken. If the CCC had acted as I suggested in '83 and '84, and had made that the ONLY change in Cobol-97, it would still have done more for the user community than with every other change it has provided—and by factor of ten. There's still time to provide a factor-of-one savings, so I hope it will be done. (Of course, file expansion can also be avoided if a four-digit year is used in a more compact form. The new binary usages will help this, but they only go part way; in order to integrate such compressed date formats into COBOL, the DATE attribute I suggested above is needed.) (I've consulted the Cutter Info. Corp.'s Surviving the Year 2000 report, and the back issue set of Tick Tick Tick, in writing the above. Some of the authorities in those publications advocate avoidance of year expansion, so the position is not unreasonable.)

Response: This will be considered as an enhancement for a future standard.

Page 249, paragraph 13.15.14.2, Rule 6, and General Rule 2 on next page—delete. (I assume someone is looking after deleting items that deal with the ATTRIBUTE clause. (Too bad; it's possible to implement it and it adds value to RW.) Other locations affected include 8.3.1.1.2.8 and 8.10.)

Response: We accept that the book is inconsistent. The full specification of the ATTRIBUTE clause will be included.

*5. Eliminate (or at least tone down) Substantive-Change-Not-Affecting #81, "The contents of a character position described with the PICTURE character 'A' are not required to be a letter". I was present in Chicago when this was passed, 5-1, and I couldn't get a good justification out of those in favor except that "it doesn't do anything and we're sick of it". My pointing out that Validate would make use of it and prove its inclusion to be far-sighted made no impression. (Fortunately General Rules 15A (292) and 17A (294) of Picture seem not to have been modified. See also General Rule 3 of USAGE, on page 336 and General Rule 3a of VALIDATE on page 497.) The user community shouldn't be given the impression, by this item, that PIC A is meaningless.

Response: Accept.

B. Editorial: Foreword & Substantive-Changes-Not-Affecting List.

Although these are only editorial changes, this is the most "public" page of the standard; it conveys "what's new, feature-wise", which is the public's main concern, via the press. Therefore it should be well-organized, error-free, and complete.

Code: 1058-74331
Ref: 1968-7378

JAPANESE PATENT OFFICE
PATENT JOURNAL (A)
KOKAI PATENT APPLICATION NO. HEI 5[1993]-27947

Int. Cl.⁵:

G 06 F 7/24
15/02

Sequence Nos. for Office Use:

8323-5B
9194-5L

Filing No.:

Hei 3[1991]-177982

Filing Date:

July 18, 1991

Publication Date:

February 5, 1993

No. of Claims:

1 (Total of 4 pages)

Examination Request:

Not filed

METHOD OF GUARANTEEING YEAR ORDER

Inventor:

Masakazu Hazama
Hitachi, Ltd., Information Systems
Development Headquarters
890-12 Kashimada, Saiwai-ku,
Kawasaki-shi, Kanagawa-ken

Applicant:

000005108
Hitachi, Ltd.
6 Surugadai, 4-chome, Kanda,
Chiyoda-ku, Tokyo-to

Agent:

Katsuo Ogawa, patent attorney

[There are no amendments to this patent.]

Purpose

Constitution

[illegible]

- Key:**
- | | |
|----|--|
| 1 | Parameter analysis processing |
| 2 | Work area input processing |
| 3 | Year evaluation processing |
| 4 | Replacement processing |
| 5 | Work area output processing |
| 6 | File |
| 7 | Program |
| 8 | Work area |
| 9 | Parameter |
| 10 | Year 2000 date correspondence utility module |
| 11 | Processing unique to program |
| 12 | File input processing |
| 13 | Module call-up processing |
| 14 | Work area name start position |

Claim

Claim

1. Method of guaranteeing year order characterized in that, in a computer system that has a memory means and a processing section, when the last 2 digits for years in the 1900's and 2000's AD are stored in the aforementioned memory means, the processing section replaces the

code for the 10's place in the last 2 digits of the year AD with a code that maintains the year order.

Detailed explanation of the invention

[0001]

Industrial application field

This invention pertains to a method of guaranteeing year order for the year 2000 AD that guarantees ascending/descending order with evaluation of magnitude and sort/merge processing using programs for digital files in which the last 2 digits of the year AD are stored, and that can obtain correct results.

[0002]

Prior art

With conventional computer systems, the last 2 digits of years AD are stored. Note that, for example, Japanese Kokai Patent Application Nos. Sho 58[1983]-1229 and Hei 03[1991]-22117 involve this type of technology.

[0003]

Problems to be solved by the invention

The aforementioned prior art does not take into consideration years after 2000 AD, and manages the year with the last 2 digits of the year AD in a file. For this reason, after 2000 AD, when ascending/descending order is handled by processing that evaluates magnitude and by sort/merge processing using normally numbered years, their relative magnitudes are represented by formula 1.

$$1999 > 1998 > 2001 > 2000$$

[0005]

That is, regardless of the fact that the year 2000 must be evaluated to be larger than the year 1999, for evaluation, only the last 2 digits of the date are used, so since 00 is smaller than 99, year 2000 is evaluated to be smaller than year 1999. Using 4 digits for the date has been considered as a method of resolving this, but in this case, it is necessary to change the data file record length and block length, and program modifications also arise.

[0006]

The purpose of this invention is to provide a method of guaranteeing year order for handling 2000 AD that makes use of the code system and that can also handle years after 2000 AD.

[0007]

Means to solve the problems

To accomplish the aforementioned purpose, in a computer system that has a memory means and a processing section, when the last 2 digits for years in the 1900's and in the 2000's AD are stored in the aforementioned memory means, the processing section will replace the code of the 10's place in the last 2 digits of the date with a code that maintains the year order.

[0008]

Function

When there are data present that indicate years in the 1900's and 2000's AD, the data code that represents the date is replaced by another code so that the year order will be maintained. In this way, magnitude evaluation and ascending/descending order processing are guaranteed.

[0009]

Application example

Figure 1 is a block diagram of a program in one application example of this invention.

[0010]

(6) is a file where data that include only the last 2 digits of the year AD are stored. (7) is a program. (8) is a clear area. (9) is a parameter. (10) is a 2000 AD correspondence utility module.

[0011]

2000 AD correspondence utility module (10) (hereafter called module (10)) is activated when specified by program (7) or a utility and is positioned as pre-processing for processing that handles the year. Note that, in module (10), a range of the last 2 digits for which code transformation will be performed are specified in advance. Replacement involves numbers for years in the 2000's where the last 2 digits are smaller than the smallest number in the last 2 digits in years in the 1900's. For example, when data in file (6) for years AD begin with the year 1973, the last 2 digits are replaced using 00 (year 2000) for 72 (year 2072). The present application example is an example where there are data from year 1960 in file (6), such a range is specified

so that the last 2 digits will be transformed to codes 00-59. Note that, in the present application example, EBCDIC code is used as the code.

[0012]

Next, the processing sequence will be explained. First, program (7) calls module (10) at a preliminary stage that evaluates the year. After that, the following processing is performed by module (10).

[0013]

Parameter analysis processing (1): parameter (9), provided from outside, is input and the contents are analyzed, and the starting positions of the work area name and the last 2 digits of the year AD in the record are confirmed.

[0014]

Work area input processing (2): data from the work area name obtained by parameter analysis processing (1) are input.

[0015]

Year evaluation processing (3): 2 bytes from the starting position of the last 2 digits in the year AD in the record obtained by parameter analysis processing (1) are evaluated, and if within a fixed range, in the present application example, in the range from '00' to '59,' replacement processing (4) is performed. In addition to this, the next data are input.

[0016]

Replacement processing (4): the 10's place in the last 2 digits in the year AD in the record is replaced as in Table 1.

[0017]

Table 1

	置換前①	置換後②
③ 十の位が0の場合	X' F 0'	X' F A'
十の位が1の場合	X' F 1'	X' F B'
十の位が2の場合	X' F 2'	X' F C'
十の位が3の場合	X' F 3'	X' F D'
十の位が4の場合	X' F 4'	X' F E'
十の位が5の場合	X' F 5'	X' F F'

Key: 1 Before replacement
 2 After replacement
 3 When 10's place is __

[0018]

Work area output processing (5): data that have undergone replacement processing (4) are output to work area (8).

[0019]

By replacing character codes as shown in Table 1, it is evaluated that year 2000 is greater than year 1990, and that year 2010 is greater than year 2000.

[0020]

Note that, in the present application example, the replacement processing with codes shown in Table 1 was performed for [years] greater than year 2000 with EBCDIK code, but in the case of [years] less than year 2000, they could also be replaced by empty code as in Table 2. With this method, for example, if there are data from year 1999 in file (6), up to year 2098 can be handled.

[0021]

Table 2

	置換前①	置換後②
③ 十の位が0の場合	X' F 0'	X' B 0'
十の位が1の場合	X' F 1'	X' B 1'
十の位が2の場合	X' F 2'	X' B 2'
十の位が3の場合	X' F 3'	X' B 3'
十の位が4の場合	X' F 4'	X' B 4'
十の位が5の場合	X' F 5'	X' B 5'
十の位が6の場合	X' F 6'	X' B 6'
十の位が7の場合	X' F 7'	X' B 7'
十の位が8の場合	X' F 8'	X' B 8'
十の位が9の場合	X' F 9'	X' B 9'

Key: 1 Before replacement
 2 After replacement
 3 When 10's place is __

[0022]

Also, for [years] less than year 2000, they could also be replaced with code that uses X'F0' → X'B0', etc.

[0023]

Also, in the case of years in the 2000's, they could be replaced by X'F0' → X'C0', and for years in the 1900's, replaced by X'F0' → X'B0'. That is, both years in the 2000's and in the 1900's could be replaced by other codes.

[0024]

Also, it makes no difference if the character code used is JIS code or ASCII code, etc. That is, when the relative magnitudes of years are evaluated, code replacement need only be performed so that evaluation is correctly accomplished.

Effect of the invention

Effect of the invention

By replacing code so that the relative magnitudes of years are correctly evaluated, the effect is that year order will be guaranteed without changing data file record length or block length, and further, without modifying programs.

Brief description of the figures

Figure 1 is a figure that shows program configuration.

Explanation of symbols

Explanation of symbols
(1) ... parameter analysis processing, (2) ... work area input processing, (3) ... year evaluation processing, (4) ... replacement processing, (5) ... work area output processing, (6) ... file, (7) ... program, (8) ... work area, (9) ... parameter, (10) ... 2000 AD correspondence utility module.

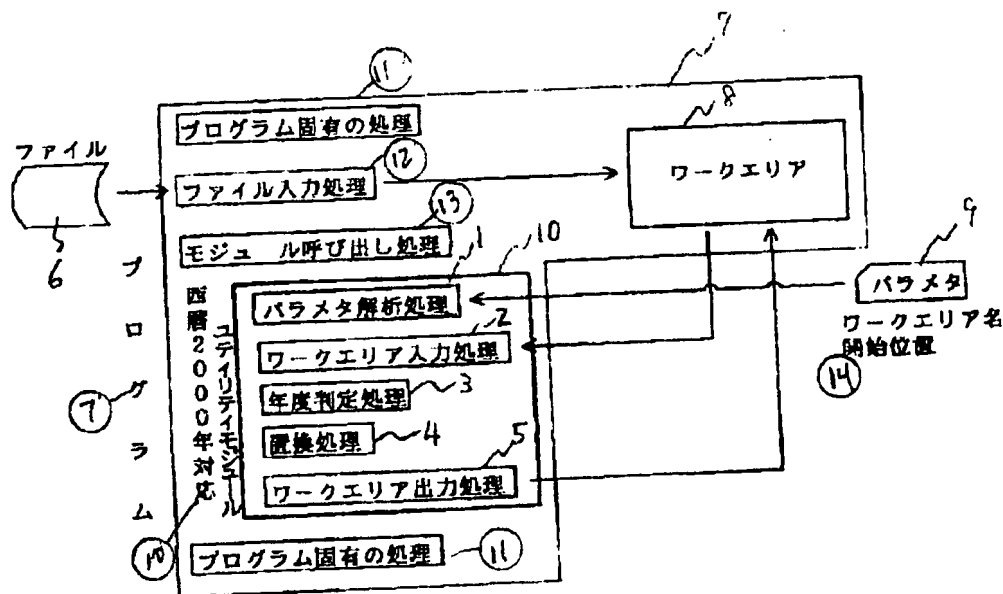


Figure 1

- Key: 1 Parameter analysis processing
2 Work area input processing
3 Year evaluation processing

- 4 Replacement processing
- 5 Work area output processing
- 6 File
- 7 Program
- 8 Work area
- 9 Parameter
- 10 Year 2000 date correspondence utility module
- 11 Processing unique to program
- 12 File input processing
- 13 Module call-up processing
- 14 Work area name start position

December 2, 1999 11:27am Page 1

? L s22/7/all

22/7/1

DIALOG(R) File 347:JAPIO

(c) 1999 JPO & JAPIO. All rts. reserv.

04036247 **Image available**
YEAR ORDER ASSURANCE METHOD

PUB. NO.: 05-027947 [JP 5027947 A]
PUBLISHED: February 05, 1993 (19930205)
INVENTOR(s): HAZAMA MASAKAZU
APPLICANT(s): HITACHI LTD [000510] (A Japanese Company or Corporation), JP
(Japan)
APPL. NO.: 03-177982 [JP 91177982]
FILED: July 18, 1991 (19910718)

ABSTRACT

PURPOSE: To assure the order of years without changing the record length and block length of a data file and changing a *program* by substituting codes so as to normally judge the relative relation of years.

CONSTITUTION: A utility module 10 corresponding to *2000* A.D. is operated when being designated by a *program* 7 or utility, and positioned as the preprocessing of a processing to handle the years. The range of lower *two* *digits* to execute code conversion is designated to the module 10 in advance. The number of the 2000s having the lower *two* *digits* smaller than the smallest number of the lower *two* *digits* in the 1900s is substituted. For example, when the year of data in a file 6 starts from the 1973 A.D., the years having the lower *two* *digits* from '00' (the *2000* A.D.) to '72' (the 2072 A.D.) are converted. Thus, relative number judgement and ascending/descending order processing are assured.

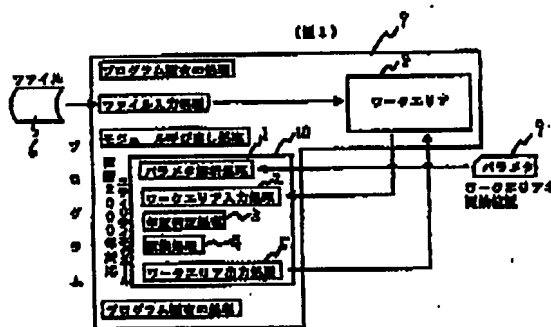
(43)公開日 平成5年(1993)2月5日

技術表示箇所

審査請求 未請求 請求項の数1(全 4 頁)

(74) 代理人 弁護士 小川 勝男

【構成】年度の大小判定、ソート・マージ処理の前に、西暦2000年対応ユーティリティモジュール10を起動する。その後、外部パラメータにより予め西暦下2ケタが格納されているレコード中の位置を指定し、予め指定した範囲の西暦を示すコードを、年度の順序が維持される様に他のコードを置き換える。



(3)

特開平5-27947

【表1】

(表1)

	置換前	置換後
十の位が0の場合	X' F 0'	X' F A'
十の位が1の場合	X' F 1'	X' F B'
十の位が2の場合	X' F 2'	X' F C'
十の位が3の場合	X' F 3'	X' F D'
十の位が4の場合	X' F 4'	X' F E'
十の位が5の場合	X' F 5'	X' F F'

【0018】ワークエリア出力処理5：置換処理4を行ったデータをワークエリア8に出力する。

【0019】表1に示される様に、文字コードを置き換えることにより、1990年よりも2000年が、2000年代より2010年が大きいと判断される。

【0020】なお、本実施例では、EBCDICコードで、2000年以上の場合、表1に示すコードの置き換*

*え処理を行ったが、2000年未満の場合に、表2の様に、空いているコードと置き換えてもよい。このやり方では、例えば、ファイル6に1999年からのデータがある場合ならば、2098年まで対応できる。

【0021】

【表2】

(表2)

	置換前	置換後
十の位が0の場合	X' F 0'	X' B 0'
十の位が1の場合	X' F 1'	X' B 1'
十の位が2の場合	X' F 2'	X' B 2'
十の位が3の場合	X' F 3'	X' B 3'
十の位が4の場合	X' F 4'	X' B 4'
十の位が5の場合	X' F 5'	X' B 5'
十の位が6の場合	X' F 6'	X' B 6'
十の位が7の場合	X' F 7'	X' B 7'
十の位が8の場合	X' F 8'	X' B 8'
十の位が9の場合	X' F 9'	X' B 9'

【0022】また、2000年未満の場合に、X' F 0' → X' C 0' 等、使用しているコードと置き換えてもよい。

【0023】また、2000年代の場合に、X' F 0' → X' C 0' と、1900年代の場合に、X' F 0' → X' B 0' と置き換えてもよい。つまり、2000年代と1900年代の両方を他のコードに置き換えてもよい。

【0024】また、使用する文字コードはJISコードやASCIIコード等でも構わない。つまり、年度の

大小関係等を判断するときに、判断が正常に行える様にコードの置き換えを行えばよい。

【0025】

【発明の効果】年度の大小関係判定が正常に行える様にコードの置き換えを行うことにより、データファイルのレコード長、ブロック長を変更せず、またプログラム修正もせずに年度の順序が保証されるという効果がある。

【0026】

【図面の簡単な説明】

【図1】プログラムの構成を示す図である。

(3)

特開平5-27947

【表1】

(表1)

	置換前	置換後
十の位が0の場合	X' F0'	X' FA'
十の位が1の場合	X' F1'	X' FB'
十の位が2の場合	X' F2'	X' FC'
十の位が3の場合	X' F3'	X' FD'
十の位が4の場合	X' F4'	X' FE'
十の位が5の場合	X' F5'	X' FF'

【0018】ワークエリア出力処理5：置換処理4を行ったデータをワークエリア8に出力する。

【0019】表1に示される様に、文字コードを置き換えることにより、1990年よりも2000年が、2000年代より2010年が大きいと判断される。

【0020】なお、本実施例では、EBCDICコードで、2000年以上の場合、表1に示すコードの置き換* (表2)

(表2)

	置換前	置換後
十の位が0の場合	X' F0'	X' B0'
十の位が1の場合	X' F1'	X' B1'
十の位が2の場合	X' F2'	X' B2'
十の位が3の場合	X' F3'	X' B3'
十の位が4の場合	X' F4'	X' B4'
十の位が5の場合	X' F5'	X' B5'
十の位が6の場合	X' F6'	X' B6'
十の位が7の場合	X' F7'	X' B7'
十の位が8の場合	X' F8'	X' B8'
十の位が9の場合	X' F9'	X' B9'

【0022】また、2000年未満の場合に、X' F0' → X' C0' 等、使用しているコードと置き換えてもよい。

【0023】また、2000年代の場合に、X' F0' → X' C0' と、1900年代の場合に、X' F0' → X' B0' と置き換えてもよい。つまり、2000年代と1900年代の両方を他のコードに置き換えてもよい。

【0024】また、使用する文字コードはJISコードやASCIIコード等でも構わない。つまり、年度の

*え処理を行ったが、2000年未満の場合に、表2の様に、空いているコードと置き換えてもよい。このやり方では、例えば、ファイル6に1999年からのデータが有る場合ならば、2098年まで対応できる。

【0021】

(表2)

大小関係等を判断するときに、判断が正常に行える様にコードの置き換えを行えばよい。

【0025】

【発明の効果】年度の大小関係判定が正常に行える様にコードの置き換えを行うことにより、データファイルのレコード長、ブロック長を変更せず、またプログラム修正もせずに年度の順序が保証されるという効果がある。

【0026】

【図面の簡単な説明】

【図1】プログラムの構成を示す図である。

(4)

特開平5-27947

5

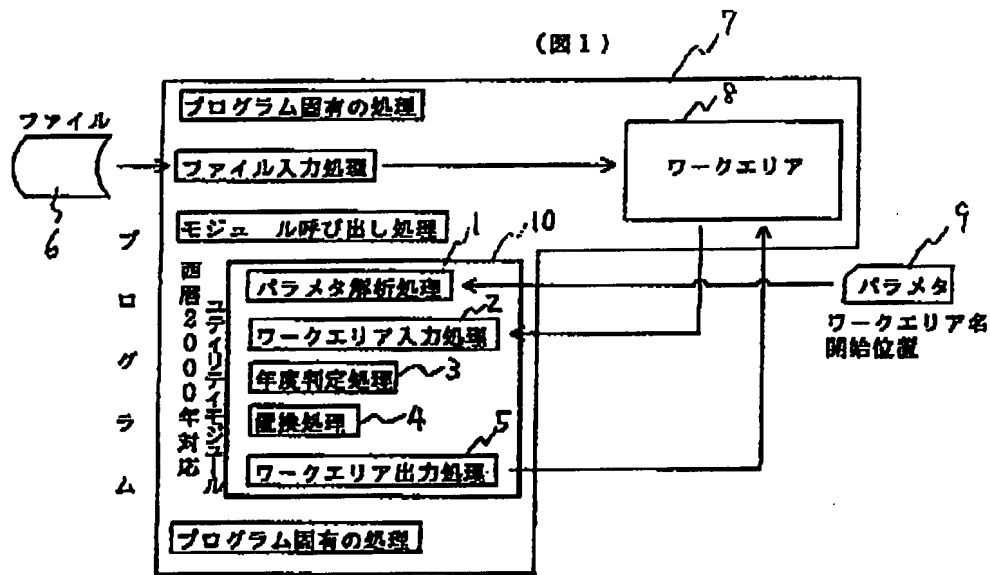
6

【符号の説明】

1…パラメタ解析処理、2…ワークエリア入力処理、3…年度判定処理、4…置換処理、5…ワークエリア出力

処理、6…ファイル、7…プログラム、8…ワークエリア、9…パラメタ、10…西暦2000年対応ユーティリティモジュール。

【図1】



around. Remind your listeners that libraries are in the business of acquiring and keeping books. And if your local library doesn't have the one you want, ask for the interlibrary loan and they'll get it for you from somewhere else fast and cheap.'

SIEGEL: Several listeners wrote after hearing about Bruce Dickens, who patented a method for coping with the Y2K computer problem. They described experiences like that of Sal Miccio of West Hurley, New York.

WERTHEIMER: Mr. Miccio says, 'The method you described is one I put into place over 30 years ago. While working for the IBM Credit Union in Poughkeepsie in 1977, we put our first 30-year mortgage on the system and it brought the computer to its knees because it couldn't figure out the maturity year of 2007. It was at that time we devised the compare to 50. If the two-position year was greater than 50, it meant the 1900s, and if less than 50, the year 2000 and greater. I've worked on many other banking and credit union systems since then and they all had the same routine, except that lately the compare was to 80 instead of 50.'

SIEGEL: Mr. Miccio continues, 'If Mr. Dickens got a patent for this method, then we pioneers who actually have been using the code for over 32 years, should be co-patent holders. If this is the method that Mr. Dickens got a patent for, then it doesn't say much for the people at the Patent Office.'

WERTHEIMER: Finally, Kenneth Huey writes, 'I do believe I caught Bob Mondello in an error. In his review of "Felicia's Journey," he attributes both "The Sweet Hereafter" and "Affliction" to the director Atom Egoyan, whereas, in fact, the common factor was Russell Banks, the author of the novels on which both movies were based. While Egoyan did indeed direct "The Sweet Hereafter," the director of "Affliction" was Paul Schrader. So do I win a mug or something?'

SIEGEL: Sorry, Mr. Huey, you would have to share the mug with several others who made the same correction.

WERTHEIMER: Our e-mail address is atc@npr.org. The street address, Letters; ALL THINGS CONSIDERED; National Public Radio; 635 Massachusetts Avenue Northwest; Washington DC 20001.

SIEGEL: And no matter which address you use, please tell us where you're writing from and how you pronounce your name.

LANGUAGE: English

LOAD-DATE: December 29, 1999

National Public Radio (NPR)

SHOW: ALL THINGS CONSIDERED (8:00 PM ET)

November 18, 1999, Thursday

HEADLINE: LETTERS FROM LISTENERS

ANCHORS: LINDA WERTHEIMER; ROBERT SIEGEL

BODY:

LINDA WERTHEIMER, host:

This is NPR's ALL THINGS CONSIDERED. I'm Linda Wertheimer.

ROBERT SIEGEL, host:

And I'm Robert Siegel. Now to some of the letters we've received recently.

From Groton, Connecticut, Robert Korr(ph) wrote in about our story on the survivors of the Bataan Death March. 'It was moving beyond words,' he says. 'Your story was one of the few reminders today that Veterans Day is about veterans, not clearance sales.'

WERTHEIMER: There were several letters about Tovia Smith's series on adoption, especially her story about the couple suing the adoption agency because their adopted twins have cerebral palsy.

SIEGEL: Phil Wald(ph) of Butte, Montana, noted the comment from the parents that they hoped their lawsuit will help to make lawsuits such as theirs unnecessary in the future. Mr. Wald writes, 'The fact of the matter is that it already is unnecessary. They've perceived that as adoptive parents, they are somehow consumers. They have a problem with the product and so they now sue the provider. When we perceive ourselves as consumers in an economy rather than as citizens in a society, our humanity suffers.'

WERTHEIMER: And Stephanie Lee(ph) in Anchorage, Alaska, asks, 'What's next? Will adoptive parents sue because their teen-ager is insufferable? Because their grown child neglects them in their golden years?'

SIEGEL: David Crosby of Indianapolis heard our story about on-demand publishing and tried searching some library databases instead. He looked for one of the books described as out of print, Lawrence Block's novel "Random Walk."

WERTHEIMER: Mr. Crosby writes, 'I found 349 locations for the first edition and 17 more for a mass-market reprint. My local public library owns a copy of the first edition. I found that the book has already been checked out, and I placed a hold. I can wait three weeks to save \$ 20. Looks to me like my local public library is still the best bargain

Year 2000 Code Inspections

Dr. John Ricketts
Year 2000 Center of Competency
IBM Global Services
2707 Butterfield Road
Oakbrook, IL 60523
630-574-4387
jaricke@us.ibm.com

Rob Gentile
Year 2000 Conversion Center
IBM Global Services
1701 North Street
Endicott, NY 13760
607-752-5378
rgentile@us.ibm.com

Brian Phoenix
Year 2000 Conversion Center
IBM Global Services
1701 North Street
Endicott, NY 13760
607-752-2534
bnp@vnet.ibm.com

Terms and Conditions of Use:

This is a Year 2000 Readiness Disclosure and Year 2000 Statement under the Year 2000 Information and Readiness Disclosure Act. Any statements made or compilations included in this document regarding entities other than IBM are republications; such statements and compilations are based on information provided by such entities and have not been independently verified by IBM.

This report is based on information gathered in the performance of a limited number of code inspections and is provided for information purposes only. By including such information in this publication, IBM does not suggest that the information is representative of the experience others may obtain.

"If you don't care about quality, you can meet any other objective."¹ This Law of Software Engineering was written long before Year 2000 entered the mainstream, but it fits an emerging syndrome: If Year 2000 quality isn't managed, the quality you get will be "whatever it happens to be when the deadline arrives."²

Ironically, the most efficient way to manage software quality is seldom used. In general, "formal code inspections are about twice as efficient as any known form of testing."³ IBM's experience demonstrates that code inspections work for Year 2000, too.

Code Inspection Process

There are generally two ways to perform Year 2000 source code inspections. The first way, rapid inspections, looks for problem areas in a large quantity of source code, then the suspected trouble spots are examined in depth. This two-step process is an efficient way to inspect large amounts of code because fifty percent or more of the code may have no date impact whatsoever. The second way, full inspections, looks in depth at everything in a small to moderate quantity of source code. Though it takes more time, it gives higher confidence that trouble spots have been found.

Under both approaches, all the code is inspected, but rapid inspections devote more effort to some code while full inspections devote approximately equal effort to all code. Regardless of the approach, re-inspections can be done after an application has received enhancement or maintenance, or after the findings of a previous inspection have been addressed.

Code inspections generally use the client's Year 2000 coding standards and exemptions. Thus, something that would be counted as a problem for one client isn't always counted as a problem for another. For example, use of a non-standard pivot year for windowing isn't a problem if an application was granted an exemption because it was converted before the standard was set. On the other hand, some things, such as missed dates, are counted as problems, regardless of the client's standards.

Code inspections can be fully or partly automated. Though fully automated inspections may be less expensive, their effectiveness is strictly limited by the capability of the tool. In contrast, semi-automated inspections combine the power of tools with experienced human inspectors who may see problems that tools alone would miss and who can eliminate many false positives that tools alone cannot resolve. Human intelligence has been the cornerstone of effective code inspections since the technique was invented.⁴ Nevertheless, since inspections are usually of much shorter duration than either conversion or testing, cost and schedule are frequently affordable. As it is with testing, there is no guarantee that all problems will be found, but inspections can reveal many Year 2000 problems, including some that are not found by typical tests.

In many cases, the code being inspected has already been inventoried and converted. It may also have been tested and returned to production, where it probably has received further maintenance or enhancement. Of these conditions, only an inventory is required before inspection.

If conversion is complete, there should be reason to believe the code is Year 2000 ready. Yet for an independent inspection to occur, someone has to be concerned enough to sponsor it. It may be that the applications are so critical that any problems are a potential hazard. Or the conversion and testing may be suspect. But if an inspection is sponsored as part of due diligence, the client might expect that no serious problems will be found. Every inspection covered in this paper, however, discovered at least one potentially serious problem, and often more.

Findings include both genuine problems and issues that may or may not represent problems. Findings are verified by the client's subject matter experts. This enables knowledge transfer back to the client, improves the inspection process itself, and eliminates the remaining false positives from the findings. If not eliminated, false positives create a distorted view of quality that unnecessarily consumes resources for re-converting and re-testing the code.

Year 2000 code inspections thus serve several purposes:

- Reveal specific deficiencies or inconsistencies in Year 2000 conversion.
- Establish a general level of confidence in conversion or testing.
- Substitute for certain tests, such as 9/9/99 as a valid date versus an end-of-file switch or reserved value.

Problem Classification

Year 2000 readiness is "the capability of a product, when used in accordance with its associated documentation, to correctly process, provide and/or receive date data within and between the 20th and 21st centuries, provided that all other products used with the product properly exchange accurate date data with it."⁵ For code inspection purposes, a Year 2000 problem is anything that prevents the code from being Year 2000 ready.

Inspections classify problems by statement type:

- *Century* statements move or add the wrong value into the century of a date.
- *Comparison* for greater than or less than generally require conversion (but comparisons for equality or inequality do not).
- *Calculations* often must be converted to prevent overflow or negative results.
- *Search* statements usually require conversion of dates in tables and search arguments.
- *Sort* (and merge) statements often must be converted if the key contains a date.
- *Other* statements can also have date problems. For example, moving a date to a shorter field may truncate the century.

Inspections also classify problems by problem type:

- *Missed dates* should have been converted, but weren't.
- *Incorrect fixes* fail to eliminate some original problems or create new ones.

Findings

Year 2000 code inspections can be done for any application type, computing platform, or language. The sample included in this paper covers business applications on MVS and AS/400 in COBOL, PL/I, Assembler, CSP, and RPG.

The dominant conversion approach prior to inspection was windowing, though expansion sometimes had been done, and compression or encoding were occasionally done instead. However, no conversion approach was immune to problems.

All the inspections were semi-automated. Every potential problem identified by tools was further evaluated by human inspectors.

Problems

The following are examples of Year 2000 problems detected by code inspections.

- Century always defaults to 19.
- Century is extracted from wrong bytes of date field. (Example: MMDDYY confused with YYMMDD.)
- Century is lost through truncation.
- Comparison of dates in different formats. (Example: CCYYDDDD vs. CC).
- Comparison of period numbers fails when dates have different centuries.
- Date routines are called before their arguments receive valid values.
- Dates without centuries are compared for "greater than" or "less than".
- Dates without centuries control loops.
- Duration calculations are invalid when dates have different centuries.
- Duration calculations are off by one in leap years.
- Leap year calculation routines are invalid.

- Merge keys include date without century.
- Multiple pivot years are used by different programs in same application.
- Multiple pivot years are used within same program.
- Pivot year implementation is inconsistent. (Example: 80 sometimes yields 1980, sometimes 2080.)
- Pivot year implementation is invalid. (Example: window is 1950-2049, but standard is 1951-2050.)
- Return codes from date routines are checked, but inappropriate actions are taken anyway.
- Return codes from date routines are not checked, so invalid values are used.
- Single non-standard pivot year is used.
- Sort keys include date without century.
- SQL WHERE clause fails on dates after 1999 because dates in parameters do not include centuries.
- Standard date routines are not used for dates on screens or reports.
- Subtraction of 1 from year yields negative result when year is 0.
- Validation of days in month allows wrong number of days.
- Windowing is done to dates with range expected to exceed 100 years. (Example: birth dates.)
- Windowing logic follows rather than precedes some comparisons or calculations.
- Windowing logic uses invalid constant. (Example: 80000 should be 800000).

These findings are notable because some of them indicate lack of conversion standards or enforcement. Standards-related problems tend to occur in enterprises that have multiple maintenance teams doing independent Year 2000 conversions along with their normal work.

Another reason for concern is the code had already been tested in most cases. Some problems are harder to detect via testing than inspection. To detect multiple pivot years via testing, for instance, test cases generally would have to cover the full range of each window, which seldom happens with production data or existing test data. Other explanations are not aging the test data or failing to review the output thoroughly.

Yet another reason for concern is the code had already been returned to production in some cases. Thus, some of the applications may be storing dates and date-dependent values, such as durations or balances, which could surface at a later time as erroneous data. As more time elapses, the more such errors may be accumulating, and the more difficult they may be to correct for business policy, regulatory, or legal reasons.

Issues

The following are examples of Year 2000 issues raised by inspections, but not initially counted as problems:

- Alpha date field is moved to a numeric date field.
- Century indicators are inconsistent across programs. (Example: 19 is indicated by both 0 and 1; 20 is indicated by both 1 and 2.)
- Comments say a program was designed to work only with a given range of dates, and tables are set up based on this range, but the range ends in the past.
- Conversion changed a date comparison from ">=" to ">".
- Data is moved between fields with compatible data types, but conflicting names. (Example: PAY-YY to PAY-DD.)
- Period symbols are changed to commas. (In COBOL, period symbols are scope terminators, but commas are ignored, so this change could have significantly changed the logic.)
- Pivot year is only one digit long.
- Two fields with same name in different records differ in length by 2 bytes, suggesting an unexpanded date.
- Unwindowed dates are used in comparisons or computations instead of their corresponding windowed versions.

The common thread joining these issues is human intelligence. The inspectors saw some unusual situations and noted them for discussion with the client's subject matter experts. Some of the issues were later determined to be legitimate problems.

Though the capability of Year 2000 tools continues to improve, there is no substitute for experienced inspectors because some issues occur so infrequently or are so arcane that no tool vendor could afford to build tools that cover them. There also is no substitute for active participation by the client's subject matter experts because only they have the domain knowledge needed to make the final determination of whether the identified problems and issues are legitimate concerns.

Metrics

The metrics reported here were gathered from Year 2000 code inspections performed for 10 clients. Since the sample is small and non-random, these metrics should not be used as benchmarks or standards. They do, however, illustrate that Year 2000 code inspections can detect some problems that escape various forms of conversion, testing, and maintenance.

Table 1 shows the Problem Classification Summary.

- Among statement types, Century and Comparison statements account for about 80% of the problems. Furthermore, this finding is consistent across individual inspections.
- Between problem types, Missed Dates are about 70%, and Incorrect Fixes are the remaining 30%. For individual inspections, however, the range is wide: Missed Dates have been as high as 100% and as low as 14%.

Table 1: Problem Classification Summary

<i>Statement Type</i>	<i>Missed dates</i>	<i>Incorrect fixes</i>	<i>Total problems</i>
Century	9%	19%	28%
Comparison	45%	9%	53%
Calculation	6%	0%	7%
Search	0%	0%	0%
Sort	2%	0%	2%
Other	8%	2%	10%
Total	69%	31%	100%

Table 2 shows the Problem Occurrence Analysis.

- Calculations are the most-frequently inspected statement type (53%), but they have one of the lowest problem rates. About 13% of date-impacted calculations have problems.
- Comparisons are the most date-impacted statement type (43%), and they have the second highest problem rate. About 32% of date-impacted comparisons have problems.
- Sorts are among the least-frequently inspected statement types (less than 1%), but they have the highest problem rate. About 90% of date-impacted sorts have problems.

Table 2: Problem Occurrence Analysis

<i>Statement Type</i>	<i>Statements Inspected</i>	<i>Statements Impacted</i>	<i>Inspected Statements with Problems</i>	<i>Impacted Statements with Problems</i>
Century	14%	29%	2%	25%
Comparison	33%	43%	1%	32%
Calculation	53%	13%	0%	13%
Search	0%	0%	0%	0%
Sort	0%	1%	14%	90%
Other	0%	15%	28%	18%

Table 3 shows Code Inspection Metrics.

- Date Density is date-impacted statements divided by total statements. The median Date Density is 3.8%, which is consistent with previously published metrics that about 3% of lines of code require Year 2000 conversion.⁶
- Problem Density is total problems divided by thousand of lines of source code (KLOC). The median Problem Density is .24, which is not as high as previously published metrics that code inspections find .5 to 1 problems per KLOC.^{7,8} One possible explanation for the difference is false positives have been removed from the metrics reported here through an inspection process that combines tools with manual inspection, while the inspection metrics published previously appear to have relied almost entirely on tools.
- Missed Date Rate is missed-date statements divided by date-impacted statements. The median Missed Date Rate is 7.2%, which is somewhat higher than the previously published estimate that about 5% of dates will be missed.⁹
- Program Impact is programs with problems divided by total programs. The median Program Impact is 15.9%, which means that the date problems tend to be concentrated in about one-sixth of each application.

Table 3: Code Inspection Metrics

<i>Metric</i>	<i>Maximum</i>	<i>Mean</i>	<i>Median</i>	<i>Minimum</i>	<i>Standard Deviation</i>
Date Density	4.6%	3.1%	3.8%	0.5%	1.89%
Problem Density	2.02	0.56	0.24	0.02	0.68
Missed Date Rate	30.5%	10.4%	7.2%	1.0%	9.5%
Program Impact	64.6%	22.0%	15.9%	3.9%	21.3%

Limitations

Though code inspections do provide insight into Year 2000 quality, they have some limitations insofar as metrics are concerned.

- Inspections focus on dates to keep cost and schedule affordable. They are not designed to detect problems in date-dependent data, such as durations or balances, because fixing date-impacted code usually fixes the dependent code, too. However, if code with problems is running in production and has populated files with erroneous date-dependent data, the problems discovered by inspection will not represent the amount of data that must be fixed.
- Design, installation, operating, and user documentation may contain problems that are not detected by code inspections.
- A statement is counted as one problem even if multiple changes must be applied to correct it. For example, if one statement has a date variable, relational operator, and a date constant that are all incorrect, this is counted as just one problem.

- Missing components are, by definition, excluded from code inspections. If any components are missing, they often include standard date routines or shared copybooks. Problems in these components affect other components, so they are significant omissions.
- In a typical portfolio of interlocking business applications, unless code inspection includes all the applications, inspections cannot examine all the interfaces between applications. Thus, some problems that exist in these interfaces (for example, incompatible pivot years applied by different applications to shared files) will probably escape detection.
- During a Year 2000 inspection there is no generally accepted way to classify problems by severity. Surprisingly few Year 2000 application problems lead to abnormal termination, but the overwhelming majority will lead to erroneous results if that part of the code is executed with the right data. Unfortunately, the impact on the enterprise and its mission cannot be assessed by code inspections.
- The inspections cover multiple languages, platforms, and conversion approaches, but there is no way to know how well the findings or metrics would apply to others.

Year 2000 is an extremely dynamic topic because needs, methods, and technology are continually changing. For example, the amount of code covered by Year 2000 inspections is rising sharply as more enterprises recognize its value. Likewise, some code inspections are being extended to cover code that has not been converted because it was not initially judged to be mission critical but now is seen as important enough to inspect. The metrics reported here are current as of the second quarter of 1999, but as conditions change, some of the metrics will probably change, too.

Lessons Learned

Despite the limitations, Year 2000 code inspections support a number of conclusions. As will be seen, the lessons learned have been translated into actions that do more than just fix the newly discovered problems.

- The 80-20 rule applies to Year 2000. Typically, about 80% of the problems occur in about 20% of the programs. Finding the right subset and focusing on it is a critical success factor.
- Some statement types are more susceptible to Year 2000 problems. Date-impacted calculations are relatively common, but they are not especially prone to problems. Although date-impacted sort statements are rare, they are quite prone to problems. Hence, devoting extra attention to the most problem-prone statement types could be another critical success factor.
- More than half the inspections found violations of the enterprise's own Year 2000 standards. Multiple pivot years are a common violation that always make subsequent maintenance and enhancement more prone to problems. And End-to-End Testing will not discover multiple pivot years unless the standards require test cases that cover window boundaries – which often requires creation of new test cases in addition to those drawn from production data or existing test beds.
- About half the inspections found problems in the construction or use of date routines. Failure to call a date routine, pass valid arguments, check the return code, or act appropriately on it defeat many of the benefits of standard date routines.

Unless dates are found accurately, conversion cannot be entirely successful. Code inspections provide another opportunity to detect dates missed previously. Furthermore, code inspections sometimes uncover evidence of multiple conversion styles within the same application. For example, there are several ways to code a date window, and some applications include more than one, even when one would do. Likewise, some conversion approaches are inherently more complicated. Century indicators, for instance, often make the code a lot more complicated than century bytes. Missed dates, consistent style, and code complexity are concerns not only for Year 2000 but for the impact on maintainability of applications thereafter.

The number of incorrect fixes found during inspections points to difficulties in testing. It's unlikely that all of the incorrect fixes detected by inspection had truly been tested with aged dates since many of the actual results would have been dramatically different from expected results. And localized testing of just changed code provides no assurance that the changes were made in the right place. For example, windowing a date after it's first used isn't detectable by a localized test of just the windowing logic. Code inspections thus provide another opportunity to detect bad fixes missed during testing.

Clean Management is standards, methodology, tools, and training that prevent loss of Year 2000 fixes and creation of new Year 2000 problems after applications return to normal maintenance and enhancement. For those applications that will be maintained and enhanced for years to come, ineffective Clean Management increases the risk that Year 2000 problems will re-emerge periodically. Code inspections provide another opportunity to detect problems created during maintenance and enhancement.

Most of the problems discovered by inspections would cause production failures at times other than the Year 2000 roll over. A few failures will be triggered on specific dates, such as those involving comparison or calculation with the current date or a particular "as of" date. But most failures could be triggered at any time – and repeatedly – by a wide range of dates occurring before, during, or after the roll over. Thus, some of the code could be producing erroneous results already, but they may not surface until well past 2000.

Abnormal program termination is the least likely consequence of many Year 2000 problems. Bad decisions and bad data are much more likely. Therefore, applications running in production without outright failures aren't necessarily running right. For example, comparisons of dates with two-digit years from different centuries will often cause incorrect logic branches rather than program termination. ("If 2001 > 1998" is true, but "if 01 > 98" is false.) Likewise, calculation with such dates may yield a result that is numerically incorrect but otherwise sound. (2001 minus 1998 gives 3, but if the result field is unsigned, as it often is in COBOL, 01 minus 98 gives 97.) The upshot of this is that undetected errors may not surface until well past 2000.

Actions Taken

When Year 2000 code inspections reveal specific deficiencies or inconsistencies in conversion, resolving issues and verifying problems are obvious next steps. Then the code must be re-converted and re-tested. Whether this should be done by the original team or vendor is a decision that often hinges on the anticipated ability of that team or vendor to do a better job in the future, as well as on the re-work. Re-inspection is an option that encourages such improvement because it subjects the re-work to another round of scrutiny.

When code inspections establish that conversion or testing in general are not achieving the level of quality required, fixing deficiencies in the process and enforcing use of the process are the most common actions. For instance, if multiple pivot dates occurred because (a) different programmers had no way to know what the standards or exemptions were or (b) the client intentionally selected multiple pivot dates, implementation of a pivot year database is one solution. On the other hand, if such a database existed but wasn't referenced, putting its use into a quality assurance checklist is another solution. In addition, adjustments in skills, tools, and standards may be necessary. For example, some enterprises now use more than one scanning tool on their code to minimize the number of missed dates. And many enterprises have begun to monitor the construction of test cases and the evaluation of test results more closely. The objective of these actions is to reduce the problem rate on subsequent work, not just on the inspected code.

When code inspections substitute for certain tests, additional action may not be required. Despite the press it gets, use of valid dates as a switch or reserved value is relatively rare. For instance, in its Gregorian format (MMDDYY), September 9, 1999 is often stored as 090999, not 999999. Likewise, April 9, 1999 is the 99th day of 1999, but in its Julian format (YYDDD), it is usually stored as 99099, not 99999. Thus, even if a value of all 9's is used as a switch or reserved value, these valid dates are not likely to be confused with it. Unless a missed date or incorrect fix related to this situation are found, no further action is required.

When code inspections reveal deficiencies in Clean Management, the necessary actions may affect people and processes not generally considered to be part of the Year 2000 project. For example, if maintenance programmers deleted or modified some Year 2000 fixes because they were unaware of the standards or the impact such changes would have on other applications, training is one solution. Of course, if no Clean Management process or tools have been implemented, this addition to the Year 2000 project is probably overdue.

Most enterprises routinely solve some number of critical production failures every month. And they divert resources from other tasks if the number of critical failures spikes during some months. One of the challenges facing enterprises as they assess their Year 2000 risk and make contingency plans is estimating how many additional failures may occur, what the mean time to repair will be, and whether the resources available to handle spikes will also be sufficient for Year 2000. Fortunately, code inspections can contribute to a general level of confidence in the quality of the Year 2000 work.

Conclusion

Far from being a simple problem with a straightforward solution, as some have suggested, Year 2000 is actually a complex set of problems with a wide variety of potential solutions. Even the best Year 2000 conversion and testing cannot remove every problem. Neither can inspections detect every problem. But based on their ability to uncover problems that escape conversion and testing, Year 2000 code inspections are increasingly being recognized as a powerful way to mitigate risk.

The Law of Software Engineering cited in the introduction says, "If you don't care about quality, you can meet any other objective." But this Law has a corollary for Year 2000: If quality is low enough to cause a large number of critical failures, no one will remember that your project was on time or within budget.

Biographies

Dr. John Ricketts is a Distinguished Engineer with IBM Global Services' Year 2000 Center of Competency. In addition to supporting estimating and metrics, he conducts special research projects, contributes to IBM's methodologies and technologies, and delivers executive education. He has received IBM Consulting Certification, IBM Technical Excellence, and Global Services awards.

Rob Gentile is Manager of IBM Conversion Centers US. He designed and led the development and implementation of several Year 2000 tools, services, and offerings. They include several for conversions and inspections. He has received IBM service, Development, and Global Services awards.

Brian Phoenix is Senior Year 2000 Architect with IBM Global Services' Year 2000 Conversion Centers. In addition to supporting the US Year 2000 conversion centers, he has designed Year 2000 analysis tools for both conversion and inspection. He has received IBM Service, Development, and Global Services awards.

¹ Gerald M. Weinberg, *Quality Software Management: First-Order Measurement*, Dorset House, New York, 1993, p. 296.

² Tom Gilb, *Principles of Software Engineering Management*, Addison-Wesley, New York, 1988, p. 328.

³ Capers Jones, *Estimating Software Costs*, McGraw-Hill, 1998, p. 511.

⁴ M.E. Fagan, "Design and Code Inspections to Reduce Errors in Program Development," *IBM Systems Journal*, 15:3, 1975, pp.182-211.

⁵ IBM, *The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation*, GC28-1251-07, 9th edition, February 1998, www.ibm.com/year2000.

⁶ Howard Rubin, "Millennium Metrics," *DCI's Year 2000 Issues & Answers Conference*, October 1996, p. D6.1.

⁷ Thomas Hoffman, "Heads Up: 'Fixed' Y2K Code Has Flaws," *Computer World*, February 23, 1999, p. 1.

⁸ Howard Solomon, "Second Opinion Y2K Insurance," *Computing Canada*, 10:25, March 12, 1999, p. 11.

⁹ Capers Jones, *The Year 2000 Software Problem: Quantifying the Costs and Assessing the Consequences*, Addison-Wesley, 1998, p. 91.

OBJECTIVE:
**ESTABLISH A BASELINE FOR NON-PATENT
LITERATURE USAGE IN DATABASE APPLICATIONS**

METHOD:
**EXAMINED MOST RECENT 300 PATENTS ISSUED IN
DATABASE CLASSES**

RESULTS:

34% NO NON-PATENT LITERATURE

**37% CITED 1 TO 4 PIECES OF NON-PATENT
LITERATURE**

**29% CITED 5 OR MORE PIECES OF NON-PATENT
LITERATURE**

	Pat No	Date	Class	NPL	Pat No	Date	Class	NPL	Pat No	Date	Class	NPL	Pat No	Date	Class	NPL
	US 6016501	20000118	707/203	0	US 6012056	20000104	707/5	1	US 6003045	19991214	707/205	4	US 5995983	19991130	707/204	0
	US 6016502	20000118	707/202	6	US 6012055	20000104	707/5	8	US 6003044	19991214	707/204	0	US 5995982	19991130	707/203	0
	US 6016503	20000118	707/104	10	US 6012054	20000104	707/104	10	US 6003043	19991214	707/203	4	US 5995981	19991130	707/202	2
	US 6016504	20000118	707/103	0	US 6012053	20000104	707/103	0	US 6003042	19991214	707/202	3	US 5995980	19991130	707/201	4
	US 6016505	20000118	707/102	0	US 6012052	20000104	707/102	0	US 6003041	19991214	707/201	3	US 5995979	19991130	707/200	14
	US 6009439	19991228	707/203	0	US 6009438	19991228	707/203	0	US 6003040	19991214	707/103	1	US 5995978	19991130	707/204	14
	US 6009439	19991228	707/104	3	US 6009438	19991228	707/104	3	US 6003039	19991214	707/103	0	US 5995977	19991130	707/204	1
	US 6009438	19991228	707/104	0	US 6009437	19991228	707/104	0	US 6003038	19991214	707/103	0	US 5995976	19991130	707/204	0
	US 6009437	19991228	707/102	10	US 6009436	19991228	707/102	10	US 6003037	19991214	707/102	3	US 5995975	19991130	707/103	3
	US 6009436	19991228	707/102	0	US 6009435	19991228	707/102	0	US 6003036	19991214	707/102	3	US 5995974	19991130	707/103	3
	US 6009435	19991228	707/100	7	US 6009434	19991228	707/100	7	US 6003035	19991214	707/102	4	US 5995973	19991130	707/103	8
	US 6009434	19991228	707/100	3	US 6009433	19991228	707/100	3	US 6003034	19991214	707/101	9	US 5995972	19991130	707/103	6
	US 6009433	19991228	707/100	1	US 6009432	19991228	707/100	1	US 6003033	19991214	707/100	0	US 5995971	19991130	707/102	12
	US 6009432	19991228	707/10	27	US 6009431	19991228	707/10	27	US 6003032	19991214	707/10	11	US 5995970	19991130	707/101	0
	US 6009431	19991228	707/10	2	US 6009430	19991228	707/10	2	US 6003031	19991214	707/10	0	US 5995969	19991130	707/100	13
	US 6009430	19991228	707/10	7	US 6009429	19991228	707/10	7	US 6003030	19991214	707/10	0	US 5995968	19991130	707/100	2
	US 6009429	19991228	707/10	0	US 6009428	19991228	707/10	0	US 6003029	19991214	707/10	15	US 5995967	19991130	707/100	0
	US 6009428	19991228	707/10	0	US 6009427	19991228	707/10	0	US 6003028	19991214	707/10	1	US 5995966	19991130	707/10	0
	US 6009427	19991228	707/10	2	US 6009426	19991228	707/10	2	US 6003027	19991214	707/10	6	US 5995965	19991130	707/10	0
	US 6009426	19991228	707/10	5	US 6009425	19991228	707/10	5	US 6003026	19991214	707/10	8	US 5995964	19991130	707/10	0
	US 6009425	19991228	707/10	0	US 6009424	19991228	707/10	0	US 6003025	19991214	707/10	0	US 5995963	19991130	707/10	0
	US 6009424	19991228	707/5	9	US 6009423	19991228	707/5	9	US 6003024	19991214	707/10	8	US 5995962	19991130	707/10	0
	US 6009423	19991228	707/5	0	US 6009422	19991228	707/5	0	US 6003023	19991214	707/10	3	US 5995961	19991130	707/10	32
	US 6009422	19991228	707/4	1	US 6009421	19991228	707/4	1	US 6003022	19991214	707/20	10	US 5995960	19991130	707/10	0
	US 6009421	19991221	707/201	7	US 6009420	19991221	707/201	7	US 6003021	19991214	707/205	0	US 5995959	19991130	707/10	0
	US 6009420	19991221	707/200	7	US 6009419	19991221	707/200	7	US 6003020	19991207	707/201	13	US 5995958	19991130	707/10	0
	US 6009419	19991221	707/104	0	US 6009418	19991221	707/104	0	US 5999947	19991207	707/203	0	US 5995957	19991130	707/2	2
	US 6009418	19991221	707/103	2	US 6009417	19991221	707/103	2	US 5999946	19991207	707/201	1	US RE36422	19991130	707/104	5
	US 6009417	19991221	707/103	0	US 6009416	19991221	707/103	0	US 5999945	19991207	707/200	3	US 5991779	19991123	707/206	7
	US 6009416	19991221	707/103	0	US 6009415	19991221	707/103	0	US 5999944	19991207	707/104	0	US 5991778	19991123	707/205	5
	US 6009415	19991221	707/101	8	US 6009414	19991221	707/101	8	US 5999943	19991207	707/104	3	US 5991777	19991123	707/205	12
	US 6009414	19991221	707/101	1	US 6009413	19991221	707/101	1	US 5999942	19991207	707/104	0	US 5991776	19991123	707/205	7
	US 6009413	19991221	707/101	0	US 6009412	19991221	707/101	0	US 5999941	19991207	707/103	0	US 5991775	19991123	707/205	0
	US 6009412	19991221	707/10	6	US 6009411	19991221	707/10	6	US 5999940	19991207	707/103	0	US 5991774	19991123	707/203	3
	US 6009411	19991221	707/10	6	US 6009410	19991221	707/10	6	US 5999939	19991207	707/102	0	US 5991773	19991123	707/203	0
	US 6009410	19991221	707/9	3	US 6009409	19991221	707/9	3	US 5999938	19991207	707/102	6	US 5991772	19991123	707/202	1
	US 6009409	19991221	707/104	4	US 6009408	19991221	707/104	4	US 5999937	19991207	707/101	0	US 5991771	19991123	707/202	26
	US 6009408	19991221	707/6	20	US 6009407	19991221	707/6	20	US 5999936	19991207	707/101	1	US 5991770	19991123	707/200	4
	US 6009407	19991221	707/5	18	US 6009406	19991221	707/5	18	US 5999935	19991207	707/101	4	US 5991769	19991123	707/104	7
	US 6009406	19991221	707/5	0	US 6009405	19991221	707/5	0	US 5999934	19991207	707/101	12	US 5991768	19991123	707/104	0
	US 6009405	19991221	707/5	12	US 6009404	19991221	707/5	12	US 5999933	19991207	707/100	2	US 5991767	19991123	707/104	0
	US 6009404	19991221	707/5	18	US 6009403	19991221	707/5	18	US 5999932	19991207	707/10	3	US 5991766	19991123	707/103	8
	US 6009403	19991221	707/5	0	US 6009402	19991221	707/5	0	US 5999931	19991207	707/10	16	US 5991765	19991123	707/102	11
	US 6009402	19991221	707/4	4	US 6009401	19991221	707/4	4	US 5999930	19991207	707/10	0	US 5991764	19991123	707/101	1
	US 6009401	19991221	707/3	0	US 6009400	19991221	707/3	0	US 5999929	19991207	707/7	0	US 5991762	19991123	707/100	3
	US 6009400	19991221	707/3	3	US 6009399	19991221	707/3	3	US 5999928	19991207	707/6	12	US 5991761	19991123	707/100	1
	US 6009399	19991221	707/3	2	US 6009398	19991221	707/3	2	US 5999927	19991207	707/5	1	US 5991760	19991123	707/100	4
	US 6009398	19991221	707/2	7	US 6009397	19991221	707/2	7	US 5999926	19991207	707/5	1	US 5991759	19991123	707/10	1
	US 6009397	19991221	707/2	0	US 6009396	19991221	707/2	0	US 5999925	19991207	707/5	1	US 5991758	19991123	707/6	0
	US 6009396	19991221	707/2	35	US 6009395	19991221	707/2	35	US 5999924	19991207	707/5	3	US 5991757	19991123	707/6	1
	US 6009395	19991221	707/6	0	US 6009394	19991221	707/6	0	US 5999924	19991207	707/4	3	US 5991757	19991123	707/3	1
Total NPL				154				233				166				229
Distribution:																
None				26												17
1				4				16								7
2				2				7								7
3				5				5								3
4				2				5								4
5				1				0								2
Over 5				10				16								2
				50				50								15
																50

Class 704 Speech signal processing

	Patent Number	Issue Date	Class subclass	Number of NPL ref cited	NPL Cited	Over 5 NPL Cited		Patent Number	Issue Date	Class subclass	Number of NPL ref cited	NPL Cited	Over 5 NPL Cited		
1	US 6016467 A	20000118	704/9	0	0	0	51	US 5995935 A	19991130	704/272	1	1	0		
2	US 6002998 A	19991214	704/9	0	0	0	52	US 5999903 A	19991207	704/271	0	0	0		
3	US 5999896 A	19991207	704/9	5	1	1	53	US 6014625 A	20000111	704/270	0	0	0		
4	US 5995922 A	19991130	704/9	3	1	0	54	US 6009397 A	19991228	704/270	1	1	0		
5	US 5995921 A	19991130	704/9	0	0	0	55	US 6009396 A	19991228	704/270	9	1	1		
6	US 5995920 A	19991130	704/9	3	1	0	56	US 6006189 A	19991221	704/270	0	0	0		
7	US 5991714 A	19991123	704/9	0	0	0	57	US 6006188 A	19991221	704/270	4	1	0		
8	US 5991713 A	19991123	704/9	0	0	0	58	US 5995934 A	19991130	704/270	0	0	0		
9	US 5991712 A	19991123	704/9	3	1	0	59	US 5995933 A	19991130	704/270	0	0	0		
10	US 5987404 A	19991116	704/9	5	1	1	60	US 5991727 A	19991123	704/270	0	0	0		
11	US 5983170 A	19991109	704/9	0	0	0	61	US 5991726 A	19991123	704/270	0	0	0		
12	US 5966686 A	19991012	704/9	5	1	1	62	US 5991725 A	19991123	704/270	3	1	0		
13	US 6014616 A	20000111	704/8	0	0	0	63	US 5987415 A	19991116	704/270	7	1	1		
14	US 5995919 A	19991130	704/8	0	0	0	64	US 5987414 A	19991116	704/270	6	1	1		
15	US 5974372 A	19991026	704/8	1	1	0	65	US 5983185 A	19991109	704/270	3	1	0		
16	US 5966685 A	19991012	704/8	7	1	1	66	US 5983184 A	19991109	704/270	0	0	0		
17	US 6009399 A	19991228	704/501	2	1	0	67	US 5983183 A	19991109	704/270	0	0	0		
18	US 6016473 A	20000118	704/500	3	1	0	68	US 5983182 A	19991109	704/270	1	1	0		
19	US 6016472 A	20000118	704/500	0	0	0	69	US 5974383 A	19991026	704/270	0	0	0		
20	US 6012031 A	20000104	704/500	0	0	0	70	US 5974382 A	19991026	704/270	1	1	0		
21	US 5999906 A	19991207	704/500	0	0	0	71	US 5970455 A	19991019	704/270	10	1	1		
22	US 5999905 A	19991207	704/500	0	0	0	72	US 5970454 A	19991019	704/269	5	1	1		
23	US 5987418 A	19991116	704/500	0	0	0	73	US 5987413 A	19991116	704/267	7	1	1		
24	US 5987417 A	19991116	704/500	0	0	0	74	US 6016471 A	20000118	704/266	3	1	0		
25	US 5983192 A	19991109	704/500	0	0	0	75	US 5991724 A	19991123	704/266	0	0	0		
26	US 5983191 A	19991109	704/500	0	0	0	76	US 6009395 A	19991228	704/264	1	1	0		
27	US 5974387 A	19991026	704/500	0	0	0	77	US 5995932 A	19991130	704/261	4	1	0		
28	US 5970461 A	19991019	704/500	8	1	1	78	US 6012028 A	20000104	704/260	2	1	0		
29	US 6014615 A	20000111	704/3	0	0	0	79	US 6006187 A	19991221	704/260	0	0	0		
30	US 5991711 A	19991123	704/3	0	0	0	80	US 6003005 A	19991214	704/260	7	1	1		
31	US 5978754 A	19991102	704/3	1	1	0	81	US 5991723 A	19991123	704/260	0	0	0		
32	US 5970460 A	19991019	704/278	0	0	0	82	US 5987412 A	19991116	704/260	5	1	1		
33	US 5983190 A	19991109	704/276	3	1	0	83	US 5983181 A	19991109	704/260	0	0	0		
34	US 5974386 A	19991026	704/276	1	1	0	84	US 5966691 A	19991012	704/260	4	1	0		
35	US 5970459 A	19991019	704/276	0	0	0	85	US 6009394 A	19991228	704/258	4	1	0		
36	US 6014626 A	20000111	704/275	8	1	1	86	US 6009393 A	19991228	704/258	1	1	0		
37	US 6012030 A	20000104	704/275	4	1	0	87	US 5991722 A	19991123	704/258	0	0	0		
38	US 6012029 A	20000104	704/275	0	0	0	88	US 5978765 A	19991102	704/258	0	0	0		
39	US 6009398 A	19991228	704/275	5	1	1	89	US 5978764 A	19991102	704/258	1	1	0		
40	US 5995936 A	19991130	704/275	0	0	0	90	US 5995931 A	19991130	704/257	0	0	0		
41	US 5983189 A	19991109	704/275	0	0	0	91	US 5991721 A	19991123	704/257	0	0	0		
42	US 5983188 A	19991109	704/275	0	0	0	92	US 5991720 A	19991123	704/256	1	1	0		
43	US 5983187 A	19991109	704/275	12	1	1	93	US 5987411 A	19991116	704/255	0	0	0		
44	US 5983186 A	19991109	704/275	0	0	0	94	US 5987410 A	19991116	704/255	0	0	0		
45	US 5974385 A	19991026	704/275	1	1	0	95	US 6006186 A	19991221	704/254	0	0	0		
46	US 5974384 A	19991026	704/275	3	1	0	96	US 5983180 A	19991109	704/254	3	1	0		
47	US 5970458 A	19991019	704/275	0	0	0	97	US 6003004 A	19991214	704/253	3	1	0		
48	US 5970457 A	19991019	704/275	2	1	0	98	US 5974381 A	19991026	704/253	0	0	0		
49	US 5970456 A	19991019	704/275	1	1	0	99	US 5970452 A	19991019	704/253	7	1	1		
50	US 5999904 A	19991207	704/272	8	1	1	100	US 6006185 A	19991221	704/251	0	0	0		
Totals														50	18

Results: 50% of Patents cited no non-patent literature
50% of Patents Did cite non-patent literature
18% of Patents cited 5 or more pieces of non-patent literature
32% of Patents cited 1 to 4 pieces of non-patent literature

Class 705 (Business)

				Number of NPL ref cited	NPL Cited	Over 5 NPL Cited					Number of NPL ref cited	NPL Cited	Over 5 NPL Cited
1	US 6016478 A	20000118	705/9	5	1	1	51	US 6009403 A	19991228	705/6	4	1	0
2	US 6009405 A	19991228	705/9	9	1	1	52	US 5982891 A	19991109	705/54	5	1	1
3	US 6006195 A	19991221	705/9	2	1	0	53	US 5978482 A	19991102	705/51	4	1	0
4	US 6003011 A	19991214	705/9	2	1	0	54	US 6012036 A	20000104	705/5	0	0	0
5	US 5999911 A	19991207	705/9	9	1	1	55	US 6003009 A	19991214	705/5	0	0	0
6	US 5995940 A	19991130	705/9	0	0	0	56	US 5987420 A	19991116	705/5	3	1	0
7	US 5987422 A	19991116	705/9	0	0	0	57	US 5978770 A	19991102	705/5	0	0	0
8	US 5974395 A	19991026	705/9	2	1	0	58	US 6014650 A	20000111	705/44	0	0	0
9	US 5963913 A	19991005	705/9	0	0	0	59	US 5991750 A	19991123	705/44	0	0	0
10	US 5960406 A	19990928	705/9	2	1	0	60	US 5991749 A	19991123	705/44	0	0	0
11	US 5960405 A	19990928	705/9	5	1	1	61	US 5987440 A	19991116	705/44	4	1	0
12	US 6012037 A	20000104	705/8	7	1	1	62	US 6014649 A	20000111	705/43	0	0	0
13	US 6006194 A	19991221	705/8	3	1	0	63	US 5995949 A	19991130	705/43	0	0	0
14	US 6006193 A	19991221	705/8	1	1	0	64	US 5987439 A	19991116	705/43	7	1	1
15	US 6003010 A	19991214	705/8	0	0	0	65	US 6012050 A	20000104	705/42	2	1	0
16	US 5991733 A	19991123	705/8	0	0	0	66	US 6003019 A	19991214	705/42	2	1	0
17	US 5991732 A	19991123	705/8	4	1	0	67	US 5970481 A	19991019	705/417	4	1	0
18	US 5978771 A	19991102	705/8	0	0	0	68	US 6006212 A	19991221	705/412	2	1	0
19	US 5974394 A	19991026	705/8	0	0	0	69	US 5974403 A	19991026	705/412	3	1	0
20	US 5974393 A	19991026	705/8	2	1	0	70	US 6009417 A	19991228	705/410	0	0	0
21	US 5974392 A	19991026	705/8	0	0	0	71	US 6009416 A	19991228	705/410	0	0	0
22	US RE36360 E	19991026	705/8	1	1	0	72	US 6006211 A	19991221	705/410	0	0	0
23	US 5970466 A	19991019	705/8	2	1	0	73	US 5999921 A	19991207	705/410	2	1	0
24	US 5960404 A	19990928	705/8	0	0	0	74	US 6014648 A	20000111	705/41	0	0	0
25	US 5987140 A	19991116	705/79	5	1	1	75	US 6012049 A	20000104	705/41	0	0	0
26	US 5991410 A	19991123	705/78	9	1	1	76	US 5995948 A	19991130	705/41	0	0	0
27	US 5991413 A	19991123	705/77	0	0	0	77	US 5991748 A	19991123	705/41	2	1	0
28	US 5987132 A	19991116	705/77	7	1	1	78	US 5991747 A	19991123	705/41	0	0	0
29	US 5974146 A	19991026	705/77	0	0	0	79	US 5987438 A	19991116	705/41	0	0	0
30	US 5995626 A	19991130	705/76	3	1	0	80	US 5963926 A	19991005	705/41	0	0	0
31	US 6016477 A	20000118	705/7	0	0	0	81	US 5978781 A	19991102	705/408	0	0	0
32	US 6014633 A	20000111	705/7	2	1	0	82	US 5960418 A	19990928	705/408	0	0	0
33	US 6009404 A	19991228	705/7	3	1	0	83	US 5983209 A	19991109	705/407	0	0	0
34	US 6006192 A	19991221	705/7	2	1	0	84	US 6006210 A	19991221	705/402	0	0	0
35	US 5999910 A	19991207	705/7	1	1	0	85	US 5995950 A	19991130	705/402	0	0	0
36	US 5987421 A	19991116	705/7	0	0	0	86	US 5987441 A	19991116	705/401	3	1	0
37	US 5983194 A	19991109	705/7	0	0	0	87	US 5974402 A	19991026	705/401	0	0	0
38	US 5974391 A	19991026	705/7	0	0	0	88	US 5963928 A	19991005	705/401	0	0	0
39	US 5970465 A	19991019	705/7	0	0	0	89	US 5963927 A	19991005	705/401	0	0	0
40	US 5966694 A	19991012	705/7	1	1	0	90	US 6016485 A	20000118	705/400	0	0	0
41	US 5963912 A	19991005	705/7	0	0	0	91	US 6014651 A	20000111	705/400	0	0	0
42	US 5963911 A	19991005	705/7	1	1	0	92	US 5999920 A	19991207	705/400	0	0	0
43	US 5963910 A	19991005	705/7	2	1	0	93	US 5960417 A	19990928	705/400	0	0	0
44	US 5991412 A	19991123	705/67	0	0	0	94	US 6006209 A	19991221	705/40	0	0	0
45	US 5991411 A	19991123	705/67	0	0	0	95	US 5999919 A	19991207	705/40	10	1	1
46	US 5963648 A	19991005	705/67	3	1	0	96	US 5991746 A	19991123	705/40	0	0	0
47	US 5974145 A	19991026	705/65	1	1	0	97	US 5983208 A	19991109	705/40	3	1	0
48	US 5999625 A	19991207	705/64	3	1	0	98	US 5978780 A	19991102	705/40	7	1	1
49	US 5991409 A	19991123	705/62	0	0	0	99	US 5974401 A	19991026	705/40	0	0	0
50	US 5974147 A	19991026	705/62	0	0	0	100	US 5963925 A	19991005	705/40	3	1	0
Totals												47	12

Results: 53% of Patents cited no non-patent literature
47% of Patents Did cite non-patent literature
12% of Patents cited 5 or more pieces of non-patent literature
35% of Patents cited 1 to 4 pieces of non-patent literature

Art Unit: 2776

M and

DETAILED ACTION

Drawings

1. This application has been filed with informal drawings which are acceptable for examination purposes only. Formal drawings will be required when the application is allowed.

Claim Rejections - 35 USC § 102

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless --

(e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371© of this title before the invention thereof by the applicant for patent.

3. Claims 1-2, 4-11, 13-18, 20-21 and 23 are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent Number 5,802,511 issued to David B. Kouchi et al.
(hereinafter Kouchi).
4. With respect to claim 1, Kouchi discloses a method of automatically accessing information from Internet address comprising the step of:
 - a) automatically accessing information at a specified Internet address (column 19, lines 46-50);
 - b) downloading the information from the specified Internet address (column 11, lines 47-49); and

Art Unit: 2776

- c) storing the information (column 2, lines 23-27).
- 5. As to claim 2, forming a connection between a system capable of accessing a remote Internet server and the remote Internet server (column 9, lines 14-16).
- 6. As to claim 4, the system is a computer system (column 9, lines 14-15).
- 7. As to claims 5, 11, 16 and 23, the computer system includes a memory device for storing the information (column 2, lines 23-27).
- 8. As to claim 6 and claim 10, erasing a previous version of the information when a more recent version of the information is obtained (same as updating: column 14, lines 53-60).
- 9. With respect to claim 7, a method of automatically accessing information from Internet address comprising the step of:
 - a) forming a connection between a system capable of accessing a remote Internet server and the remote Internet server (column 9, lines 14-16);
 - b) automatically accessing information at a specified Internet address (column 19, lines 46-50);
 - c) downloading the information from the specified Internet address (column 11, lines 47-49).
- 10. As to claim 8, storing the information (column 2, lines 23-27).
- 11. As to claim 9, providing the information to a user (column 8, lines 9-14).
- 12. With respect to claim 13, an apparatus for automatically accessing information from Internet addresses and providing the information to a user comprising:

Art Unit: 2776

a) a connection device configured for coupling to an Internet server for forming a connection between the apparatus and the Internet server (column 9, lines 14-16); and

b) a controller coupled to the connection device for controlling the operation of the connection device to automatically form the connection between the apparatus and the Internet server and download information through the connection device from specified Internet address (column 19, lines 46-50 and column 11, lines 47-49).

13. As to claim 14, the connection device is a modem (column 9, lines 14-16).

14. As to claim 15, the connection is formed over a telephone line (column 9, lines 14-16).

15. As to claims 17 and 21, a display device for displaying the information (column 8, lines 9-10 and Figure 10, number 1018).

16. The elements of claim 18 are rejected in the analysis above in claim 1 and this claim is rejected on that basis.

17. With respect to claim 20, a computer system for automatically accessing information from Internet addresses and providing the information to a user comprising:

a) a modem configured for coupling to a telephone line for forming a connection between the computer system and the Internet server (column 9, lines 14-16); and

b) a controller coupled to the modem for controlling the modem to automatically form the connection between the computer system and the Internet server and download information from specified Internet address (column 19, lines 46-50 and column 11, lines 47-49);
and

Art Unit: 2776

c) a storage device coupled to the modem for storing the information (column 2, lines 23-27).

Claim Rejections - 35 USC § 103

18. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

19. Claims 3, 12, 19 and 22 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent Number 5,802,511 issued to Kouchi et al. as applied to claims above, and further in view of U.S. Patent Number 5,694,546 issued to Richard R. Reisman (hereinafter Reisman).

20. As to claims 3, 12, 19 and 22, Kouchi discloses automatically accessing and downloading are performed periodically at predetermined intervals (column 19, lines 46-50 and column 11, lines 47-49).

Kouchi does not explicitly indicates downloading are performed periodically, but Reisman shows importing updates of information or information processing products, such as periodically issuing literature, or software upgrades (column 11, lines 55-57).

Art Unit: 2776

It would have been obvious to a person of ordinary skill in the computer art at the time the invention was made to modify the method of Kouchi with the teaching of Reisman to enable an information product vendor to supply an automated, or unattended, update and backup information to a mass market of computer users without the complexcity and expense of proprietary network.

Art Unit: 2776

Contact Information

Direct inquiries concerning this communication should be directed to Shahid Alam whose telephone number is (703) 305-2358. The examiner can normally be reached on Monday-Thursday from 8:00 AM to 4:30 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Michael Razavi, can be reached at (703) 305-4713.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 308-9051, (for formal communications intended for entry)

or:

(703) 305-9724 or (703) 308 6606 (for informal or draft communications, please label "PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington, Virginia, Sixth Floor (Receptionist).

Inquiries of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-9600.

Shahid Alam

October 21, 1998

Art Unit: 2776

DETAILED ACTION

Response to Amendment

1. This action is in response to the communication filed on February 1, 1999.
2. Applicant's arguments with respect to claims 1-23 have been considered but are moot in view of the new ground(s) of rejection and therefore, claims 1-23 are remained rejected.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless --

(e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371© of this title before the invention thereof by the applicant for patent.

4. Claims 1-2, 4-11, 13-18, 20-21 and 23 are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent No. 5,737,619 issued to David Judson (hereinafter Judson).
5. With respect to claim 1, Judson discloses a method of automatically accessing information from Internet address comprising the step of:

automatically accessing web page information at a specified Internet address, downloading the web page information from the specified Internet address and storing the web page information in order to allow a user to access the information while not connected to the specified Internet address (column 10, lines 12-27).

Art Unit: 2776

6. As to claim 2, forming a connection between a system capable of accessing a remote Internet server and the remote Internet server (column 3, lines 60-67).
7. As to claim 4, the system is a computer system (col. 1, lines 11-14 and col. 4, lines 52-55).
8. As to claims 5, 11 and 23, the computer system includes a memory device for storing the web page information (column 4, lines 32-40).
9. As to claim 6 and claim 10, erasing a previous version of the information when a more recent version of the web page information is obtained (same as updating: column 9, lines 2-11).
10. With respect to claim 7, a method of automatically accessing web page information from Internet address comprising the step of:

forming a connection between a system capable of accessing a remote Internet server and the remote Internet server (column 3, lines 60-67);

automatically accessing web page information at a specified Internet address and downloading the web page information from the specified Internet address to the system (column 10, lines 17-27).
11. As to claim 8, the step of storing the web page information (column 6, lines 10-13).
12. As to claim 9, the step of providing the web page information to a user (col. 6, lines 5-28).
13. With respect to claim 13, an apparatus for automatically accessing web page information from Internet addresses and providing the web page information to a user comprising:

a connection device configured for coupling to an Internet server for forming a connection between the apparatus and the Internet server (column 3, lines 60-67); and

Art Unit: 2776

a controller coupled to the connection device for controlling the operation of the connection device to automatically form the connection between the apparatus and the Internet server and download web page information through the connection device from specified Internet address (column 7, lines 46-57).

14. As to claim 14-17 and 21, the connection device is a modem, the connection is formed over a telephone line, a memory device coupled to the connection device for storing the information and a display device for displaying the web page information (column 4, lines 23-51, column 7, lines 46-47 and Figure 2).

15. As to claim 18, means for programming coupled to the controller for entering the specified Internet addresses (column 4, lines 1-7).

16. With respect to claim 20, a computer system for automatically accessing web page information from Internet addresses and providing the information to a user comprising:

a modem configured for coupling to a telephone line for forming a connection between the computer system and the Internet server; a controller coupled to the modem for controlling the modem to automatically form the connection between the computer system and the Internet server and download web page information from specified Internet address; and a storage device coupled to the modem for storing the web page information (column 4, lines 32-51 and column 8, lines 34-40).

Art Unit: 2776

Claim Rejections - 35 USC § 103

17. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

18. Claims 3, 12, 19 and 22 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent Number 5,737,619 issued to Judson as applied to claims above, and further in view of U.S. Patent Number 5,694,546 issued to Richard R. Reisman (hereinafter Reisman).

19. As to claims 3, 12, 19 and 22, Judson discloses automatically accessing and downloading are performed periodically at predetermined intervals (column 9, lines 12-23).

Judson does not explicitly indicates downloading are performed periodically, but Reisman shows importing updates of information or information processing products, such as periodically issuing literature, or software upgrades (column 11, lines 55-57).

It would have been obvious to a person of ordinary skill in the computer art at the time the invention was made to modify the method of Judson with the teaching of Reisman to enable an information product vendor to supply an automated, or unattended, update and backup information to a mass market of computer users without the complexity and expense of proprietary network.

Art Unit: 2776

Conclusion

20. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Art Unit: 2776

Contact Information

Direct inquiries concerning this communication should be directed to Shahid Alam whose telephone number is (703) 305-2358. The examiner can normally be reached on Monday-Thursday from 8:00 AM to 4:30 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Michael Razavi, can be reached at (703) 305-4713.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks, Washington, D.C. 20231

or faxed to:

(703) 308-9051, (for formal communications intended for entry) ***or:***

(703) 305-9724 or (703) 308 6606 or (703) 308-5403 (for informal or draft communications, please label "PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington, Virginia, Sixth Floor (Receptionist).

Inquiries of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-9600.

Shahid Alam

Patent Examiner

Art Unit 2776

January 6, 2000

Art Unit: 2777

DETAILED ACTION

Response to Arguments

1. This action is in response to the communication filed on May 14, 1999.
2. Applicant's request for reconsideration of the finality of the rejection of the last Office action is persuasive and, therefore, **the finality of that action is withdrawn.**

Allowable Subject Matter

3. After a further search and a through examination and in light of the prior art made of record, in light of the Applicant's argument and IDS submitted (paper number 5 and 6), claims 1-23 are allowed.

4. The following is an examiner's statement of reasons for allowance:

The method taught by the prior art of record locally stores, retrieves and outputs information objects to reduce the waiting time normally associated with the download of hypertext documents having high resolution graphics. The prior art also teaches that an information object is associated with an activated link and is displayed either when a hypertext document at the link id being downloaded or while the user is browsing the current page at the link.

Art Unit: 2777

The prior art of record neither teaches nor fairly suggest the combination elements as recited in claims 1, 7, 13 and 20 and more specially including the steps of automatically accessing, downloading and storing the web page information in order to allow a user to access the web page information while not connected to the specified Internet address.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

5. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Shahid Alam whose telephone number is (703) 305-2358.

Shahid Alam

January 6, 2000



US005978807A

United States Patent [19][11] **Patent Number:** **5,978,807****Mano et al.**[45] **Date of Patent:** **Nov. 2, 1999**

[54] **APPARATUS FOR AND METHOD OF
AUTOMATICALLY DOWNLOADING AND
STORING INTERNET WEB PAGES**

0 331 442 A2 9/1989 European Pat. Off. G06F 15/24
0 525 947 A1 2/1993 European Pat. Off. G07G 1/00
1 503 804 3/1978 United Kingdom G06F 11/00

[75] **Inventors:** Yoshizumi Mano, Cupertino; Chenchu
Chilamakuri, Fremont; Hisato Shima,
Saratoga, all of Calif.

[73] **Assignees:** Sony Corporation, Tokyo, Japan; Sony
Electronics, Inc., Park Ridge, N.J.

[21] **Appl. No.:** 08/941,583

[22] **Filed:** Sep. 30, 1997

[51] **Int. Cl.⁶** G06F 17/30

[52] **U.S. Cl.** 707/10; 707/2; 707/104;
707/500; 707/513

[58] **Field of Search** 707/2, 10, 104,
707/500, 513

[56] **References Cited**

U.S. PATENT DOCUMENTS

3,704,363	11/1972	Salmassy et al.	235/153
4,644,532	2/1987	George et al.	370/94
5,237,507	8/1993	Chasek	364/464.04
5,682,330	10/1997	Seaman et al.	707/104
5,694,546	12/1997	Reisman	395/200.9
5,694,547	12/1997	Subramanian et al.	395/200.11
5,737,619	4/1998	Judson	707/500
5,764,910	6/1998	Shachar	395/200.53
5,768,119	6/1998	Havekost et al.	364/133
5,774,667	6/1998	Garvey et al.	395/200.52
5,781,909	7/1998	Logan et al.	707/200
5,802,511	9/1998	Kouchi et al.	707/2
5,842,027	11/1998	Oprescu et al.	395/750.01

FOREIGN PATENT DOCUMENTS

0 280 020 A3 8/1988 European Pat. Off. G06F 15/74

OTHER PUBLICATIONS

IEEE, "1395-1995 Standard for a High Performance Serial Bus," 1995, USA.

Schilit, Bill N., "TeleWeb: Loosely connected access to the World Wide Web", Computer Networks and Isdn Systems, vol.: 28, Issue: 11, May 1996 pp. 1431-1444.

Primary Examiner—Paul R. Lintz

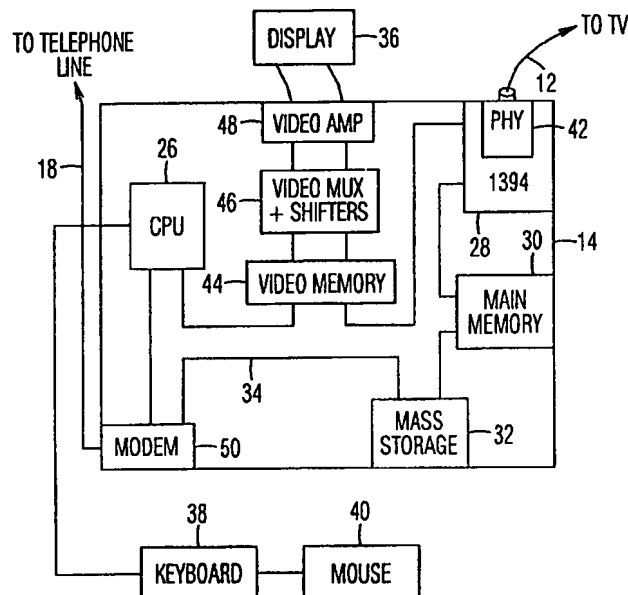
Assistant Examiner—Shahid Alam

Attorney, Agent, or Firm—Haverstock & Owens LLP

[57] **ABSTRACT**

A computer system or other internet access device is programmed to automatically access specified web pages periodically and download the information from the web page to the computer system. Through a user interface, a user programs the computer system by entering an internet address of the web page and an access interval. The access interval is the interval at which versions of the web page will be downloaded; e.g. hourly, daily or weekly. On that periodic basis the computer system or other internet access device then automatically accesses the specified web page and downloads the available information. This information is stored within a memory device associated with the computer system. The user can then access the downloaded information and view the web page without connecting to the internet to determine if there is anything of interest to the user on the web page. If there is information of interest and the user would like to obtain additional information, the user can then access the appropriate web page at a convenient time to obtain the additional information.

23 Claims, 3 Drawing Sheets



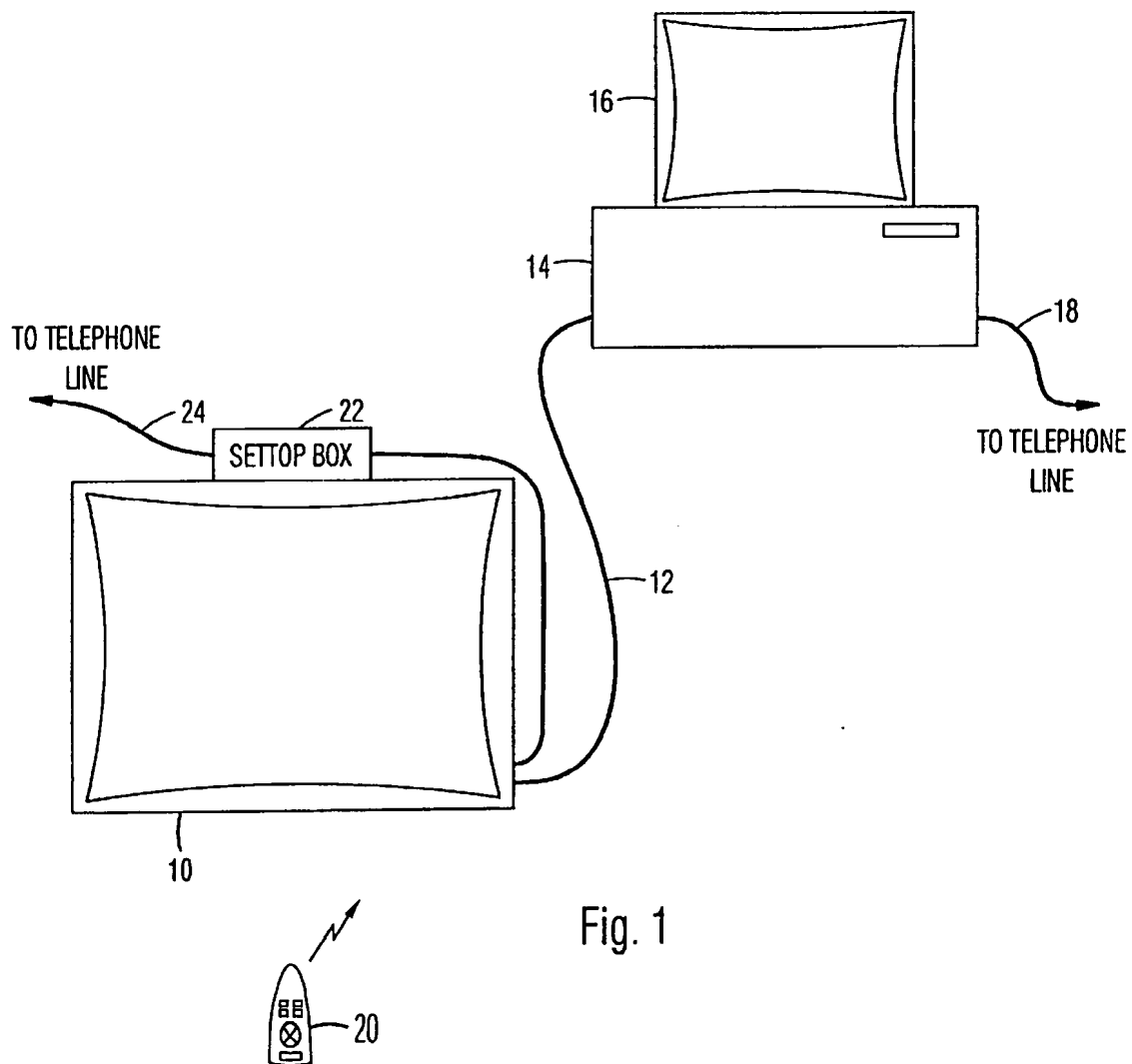


Fig. 1

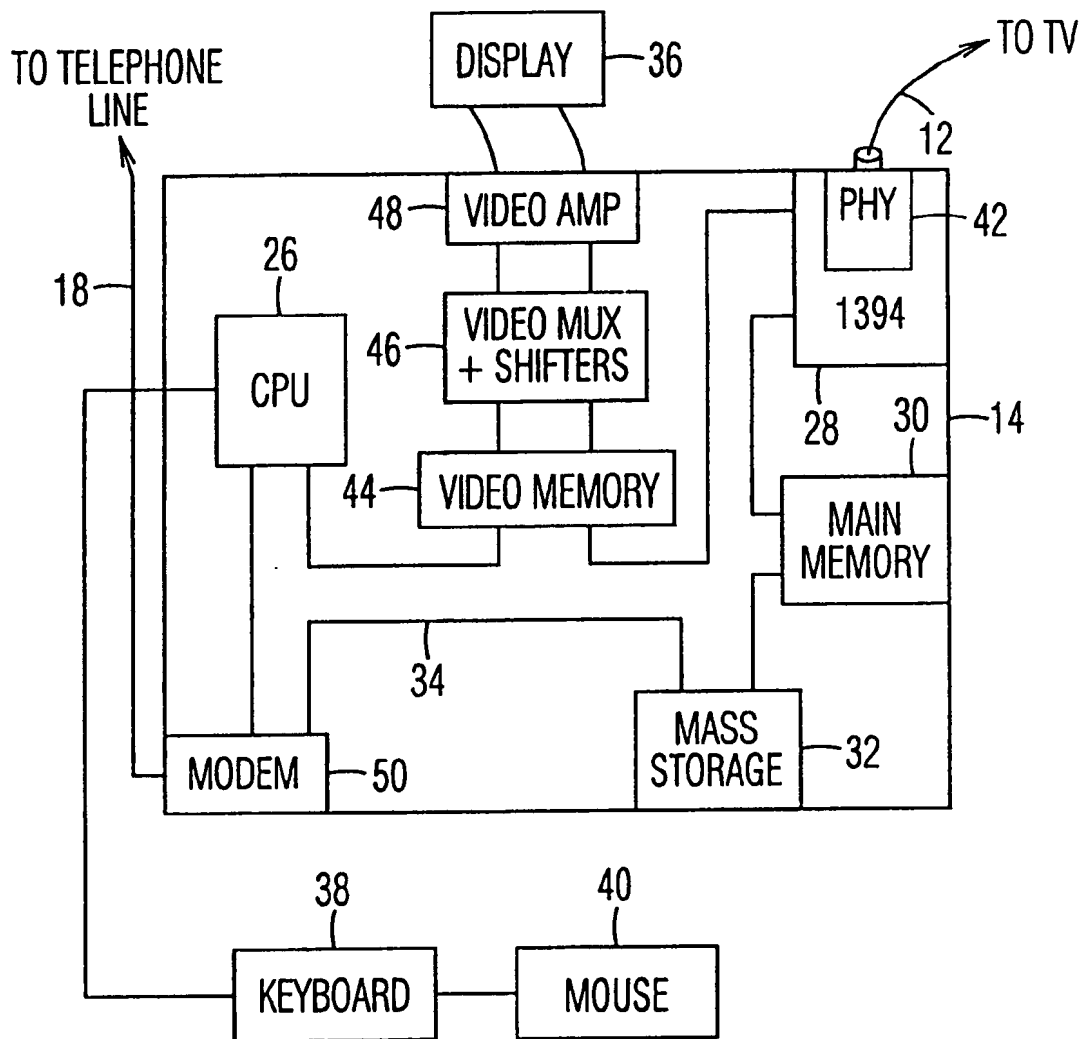


Fig. 2

60

FAVORITE ADDRESSES				
<u>ADDRESS</u> 62	<u>NICKNAME</u> 64	<u>INTERVAL</u> 66	<u>LAST UPDATED</u> 68	<u>VIEWED</u> 70
SONY.COM	SONY	DAY	3-6-97; 12:00 AM	YES 72
STOCKS.COM	QUOTES	HOUR	3-6-97; 10:00 AM	NO 74

Fig. 3

APPARATUS FOR AND METHOD OF AUTOMATICALLY DOWNLOADING AND STORING INTERNET WEB PAGES

FIELD OF THE INVENTION

The present invention relates to the field of acquiring information from the internet or world wide web. More particularly, the present invention relates to the field of using a computer system to automatically acquire information from the internet or the world wide web.

BACKGROUND OF THE INVENTION

An abundance of information is now available to users of the internet or world wide web. Sometimes the amount of available information is overwhelming to users. However, even with the wealth of information available, users of the internet often routinely return to their favorite sites and web pages.

Access to the internet and world wide web can be slow and time consuming, especially during the most popular hours of the day. Access can be much faster during the off-peak hours when not as many users are accessing the system. Even during the off-peak hours it can be time consuming to access a web page or internet site to determine if it has been updated or if there is any new information of interest to a user.

When accessing an internet site, a user instructs the computer to dial up the server of the user's internet service provider. The computer or settop box then controls the operation of a modem to establish the connection with the internet service provider. Once a connection has been made between the modem and the internet service provider, the user must then log on to the service, usually by entering a username and a password. When the user is logged on to the service, the user can then access services and information provided by the service provider and also information available through web pages at other addresses on the internet. When accessing information available through the internet, the user connects through their service provider to other servers which are providing information. This information is usually provided at internet sites and web pages. Each internet site and web page has a particular address through which it can be accessed. By entering this address, the user is instructing their internet service provider to connect them to that address.

Each internet site or web page typically has information about a certain subject. For example, an internet site provided by a newspaper will typically have current news, stories and other information provided by the newspaper. Other sites might have news, information and stock quotes about particular companies or types of companies. Still another site might have information related to a particular type of automobile. A user desiring to access such a site in order to discover whether there is any new information available since the last time the user accessed the site or any information of interest to the user, must perform the entire log on process and wait while the site is accessed through the service provider. For a user interested in many different subjects, this can be a very time-consuming process.

There are currently services which will automatically conduct a search and provide a user with information about a particular subject which is available on the internet. Typically, to use such a service, a user will enter the specific subject matter about which they would like to obtain information. This service then automatically searches the internet for the user and obtains information about the specific

subject matter. This information will be gathered by the search service from many different sites on the internet. The user is then provided with a report outlining this information and the sites from where it was gathered.

The IEEE 1394 standard, "P1394 Standard For A High Performance Serial Bus," Draft 8.02v2, Jul. 7, 1995, is an international standard for implementing an inexpensive high-speed serial bus architecture which supports both asynchronous and isochronous format data transfers. The IEEE 1394 standard provides a high-speed serial bus for interconnecting digital devices thereby providing a universal I/O connection. The IEEE 1394 standard defines a digital interface for the applications thereby eliminating the need for an application to convert digital data to analog data before it is transmitted across the bus. Correspondingly, a receiving application will receive digital data from the bus, not analog data, and will therefore not be required to convert analog data to digital data. An 'application' as used herein will refer to either an application or a device driver.

The cable specified by the IEEE 1394 standard is very thin in size compared to many other cables, such as conventional co-axial cables, used to connect such devices. Devices can be added and removed from an IEEE 1394 bus while the bus is active. If a device is so added or removed the bus will then automatically reconfigure itself for transmitting data between the then existing nodes. A node is considered a logical entity with a unique address on the bus structure. Each node provides an identification ROM, a standardized set of control registers and its own address space.

What is needed is a system which automatically obtains information from user specified internet sites. What is further needed is a system which automatically obtains information from specific internet sites during a specified time period, while a user is not using the system.

SUMMARY OF THE INVENTION

A computer system or other internet access device is programmed to automatically access specified web pages periodically and download the information from the web page to the computer system. Through a user interface, a user programs the computer system by entering an internet address of the web page and an access interval. The access interval is the interval at which versions of the web page will be downloaded; e.g. hourly, daily or weekly. On that periodic basis the computer system or other internet access device then automatically accesses the specified web page and downloads the available information. This information is stored within a memory device associated with the computer system. The user can then access the downloaded information and view the web page without connecting to the internet to determine if there is anything of interest to the user on the web page. If there is information of interest and the user would like to obtain additional information, the user can then access the appropriate web page at a convenient time to obtain the additional information.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates a block diagram of an IEEE 1394 serial bus network including a computer system and a television with a settop box.

FIG. 2 illustrates a block diagram of the relevant components within the computer system of FIG. 1.

FIG. 3 illustrates a user interface page through which the user enters the addresses of the web pages to be automatically accessed by the computer system.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

An apparatus for and method of automatically downloading and storing web pages allows a user to program a computer system or other internet access device to automatically access specified web pages periodically and download the information from the web page to the computer system. The user can then later look at the downloaded information to determine if there is anything of interest to the user on the specified web pages thus saving the user the time required to actually gain access to the web page on the internet. If there is information of interest and the user would like to obtain additional information, the user can then access the appropriate web page at a convenient time to obtain the additional information. This allows the user to review the downloaded information at their computer or other internet access device and make a decision on whether or not the new information on that web page is of interest to them without waiting through the delays associated with actually accessing the web page on the internet.

A block diagram of an IEEE 1394 serial bus network including a computer system and a television with an associated settop box is illustrated in FIG. 1. The computer 14 includes an associated display 16. The computer 14 is coupled to a telephone line 18. The computer 14 is also coupled to the television 10 through the IEEE 1394 serial bus network 12. The television is coupled to the settop box 22. The television 10 includes an associated remote control and input device 20. Preferably, the remote control device 20 utilizes infrared technology in order to send communications to the television 10. The settop box 22 is coupled to a telephone line 24.

A block diagram of the internal components of the computer 14 is illustrated in FIG. 2. While the method of automatically obtaining specified web pages can be performed on any appropriate computer system or internet access device, an exemplary computer system 14 is illustrated in FIG. 2. The computer system 14 includes a central processor unit (CPU) 26, a main memory 30, a video memory 44, a mass storage device 32, a modem 50 and an IEEE 1394 interface circuit 28, all coupled together by a conventional bidirectional system bus 34. The interface circuit 28 includes the physical interface circuit 42 for sending and receiving communications on the IEEE 1394 serial bus. The physical interface circuit 42 is coupled to the television 10 over the IEEE 1394 serial bus cable 12. In the preferred embodiment of the present invention, the interface circuit 28 is implemented on an IEEE 1394 interface card within the computer system 14. However, it should be apparent to those skilled in the art that the interface circuit 28 can be implemented within the computer system 14 in any other appropriate manner, including building the interface circuit onto the motherboard itself. The modem 50 is coupled to the telephone line 18 for sending and receiving communications over the telephone line 18. The mass storage device 32 may include both fixed and removable media using any one or more of magnetic, optical or magneto-optical storage technology or any other available mass storage technology. The system bus 34 contains an address bus for addressing any portion of the memory 30 and 44. The system bus 34 also includes a data bus for transferring data between and among the CPU 26, the main memory 30, the video memory 44, the mass storage device 32, the modem 50 and the interface circuit 28.

The computer system 14 is also coupled to a number of peripheral input and output devices including the keyboard

38, the mouse 40 and the associated display 36. The keyboard 38 is coupled to the CPU 26 for allowing a user to input data and control commands into the computer system 14. A conventional mouse 40 is coupled to the keyboard 38 or computer system 14 for manipulating graphic images on the display 36 as a cursor control device in a conventional manner. The display 36 displays video and graphical images generated by the computer system 14.

A port of the video memory 44 is coupled to a video multiplex and shifter circuit 46, which in turn is coupled to a video amplifier 48. The video amplifier 48 drives the display 36, when it is being used. The video multiplex and shifter circuitry 46 and the video amplifier 48 convert pixel data stored in the video memory 44 to raster signals suitable for use by the display 36.

A user of the system uses the keyboard 38 and the mouse 40 to program the addresses of the web pages which are to be automatically downloaded by the computer system 14. A user interface through which the user programs the web page addresses is illustrated in FIG. 3. The user interface 60 includes an address column 62, a nickname column 64, an interval column 66, an update column 68 and a viewed column 70. The user programs an entry, displayed as a row in the table, using the keyboard 38 and the mouse 40 in a conventional manner. In the address column 62, the user enters the internet address of a web page which is to be automatically downloaded. In the nickname column 64, the user assigns a nickname to correspond to this address. In the interval column 66, the user assigns an interval at which the web page is to be automatically downloaded. Acceptable entries within the interval column 66 are hour, day, number of days and week. In the last updated column 68, the computer system 14 displays the date and time of the last version of the web page which was downloaded corresponding to this entry. In the viewed column 70, the computer system 14 displays either a "Yes" or "No" value, corresponding to whether or not the last downloaded web page corresponding to this entry was viewed by the user.

The user interface example illustrated in FIG. 3 includes two entries 72 and 74. The first entry 72 includes an internet address of sony.com in the address column 62, an assigned nickname of Sony in the nickname column 64 and an interval of every day in the interval column 66. The update column 68 indicates that the corresponding web page was last updated on Mar. 6, 1997 at 12:00 AM. The viewed column 70 indicates that the user has viewed the last copy of the web page which was downloaded. In response to this first entry 72, the computer system 14 will then access the web page at the internet address sony.com every day and download the available information from that page, updating the information within the update column 68 and the viewed column 70, accordingly.

The second entry 74 includes an internet address of stocks.com in the address column 62, an assigned nickname of Quotes in the nickname column 64 and an interval of every hour in the interval column 66. The update column 68 indicates that the corresponding web page was last updated on Mar. 6, 1997 at 10:00 AM. The viewed column 70 indicates that the user has not viewed the last copy of the web page which was downloaded. In response to this second entry 74, the computer system 14 will then access the web page at the internet address stocks.com every hour and download the available information from that page, updating the information within the update column 68 and the viewed column 70, accordingly.

After the user has programmed at least one entry into the user interface 60, the computer system 14 (FIG. 2) will then

retrieve the specified web page or pages at the specified intervals. For example, in response to the first entry 72 (FIG. 3), the computer system 14 (FIG. 2) will on a daily interval, use the modem 50 (FIG. 2), to connect to the user's internet service provider through the telephone line 18 (FIG. 2). Once connected to the internet service provider, the computer system 14 (FIG. 2) will then automatically enter the address sony.com corresponding to the first entry 72 (FIG. 2). After being connected to the web page at that address by the internet service provider, the computer system 14 (FIG. 2) will download the available information from that web page, essentially making a reproduction of the information within the web page. The available information from that web page is downloaded through the telephone line 18 (FIG. 2) and the modem 50 (FIG. 2) and preferably stored on the mass storage device 32 (FIG. 2) within the computer system 14 (FIG. 2). Alternatively, the available information from that web page is stored in the main memory 30 (FIG. 2). Preferably, as a downloaded web page is updated by the computer system 14 (FIG. 2), only the latest version of that downloaded web page is saved in the mass storage device 32 (FIG. 2). Alternatively, each downloaded version of the web page can be saved until it is viewed by the user. In this alternative embodiment, after the web page is viewed by the user it will then be erased from the computer system's 14 (FIG. 2) memory.

After a web page is downloaded, a copy of the web page is in the memory of the computer 14 (FIG. 2) and available to the user. The user can then access that previously downloaded web page by opening the user interface 60 (FIG. 3) and using the mouse 40 (FIG. 2) or other input device to select the corresponding entry. If the user is interested in viewing the last downloaded web page corresponding to the first entry 72 (FIG. 3), then the user opens the user interface 60 (FIG. 3) and selects the first entry 72 (FIG. 3). The computer system 14 (FIG. 2) will then load the available information for that web page from the mass storage device 32 (FIG. 2) into the main memory 30 (FIG. 2) and display the web page on the display 36 (FIG. 2). The user can then access the available information on the downloaded web page as if they were actually accessing the original web page. However, the user will not be able to utilize any links to other internet addresses included within the web page, because the computer system 14 (FIG. 2) is not actually connected to the internet through the internet service provider. If there is any information in the web page which is interesting to the user, the user can then instruct the computer system 14 (FIG. 2) to connect to the internet service provider and view the actual web page. Once connected to the actual web page through their internet service provider, the user then has the ability to automatically jump to any links included within the actual web page.

A user who only interested in the actual information included on the downloaded web page, can save a great amount of time by viewing the downloaded web page and not waiting through the delays associated with accessing the original web page on the internet. Using the system of the present invention, the user allows the computer system 14 (FIG. 2) to automatically perform the time consuming task of accessing and downloading the specified web page. The user can then quickly open the downloaded web page and view it in a fraction of the time it would take to access the web page and view it over the internet. If the user then decides that they would like to connect to the actual web page, based on the downloaded information, they have still saved a lot of time by viewing the downloaded web page to determine if there is anything on the actual web page that is

of interest. The user will save even more time, when all the information of interest is included on the downloaded web page which has been automatically obtained by the computer system 14 (FIG. 2).

While preferably, the computer system 14 (FIG. 2) is used to enter the appropriate information into the user interface 60 (FIG. 3) and automatically download specified web pages from the internet, it should be apparent to those skilled in the art that the combination of the settop box 22 (FIG. 1), television 10 (FIG. 1) and remote control input device 20 (FIG. 1), or any other appropriate internet access device, can also be used to perform the tasks associated with the present invention. In this manner, both the computer system 14 (FIG. 2) and the television 10 (FIG. 1) with settop box 22 (FIG. 1) can be used independently to automatically download specified web pages from the internet. Alternatively, the computer system 14 (FIG. 2) can be used to automatically download the specified web pages as described above and the information accessed by the user through the television 10 (FIG. 1). In this embodiment, the television obtains the downloaded information from the computer system 14 (FIG. 2) via the IEEE 1394 serial bus 12 (FIG. 1). This downloaded information is then displayed for the user on the television 10 (FIG. 1).

The present invention has been described in terms of specific embodiments incorporating details to facilitate the understanding of principles of construction and operation of the invention. Such reference herein to specific embodiments and details thereof is not intended to limit the scope of the claims appended hereto. It will be apparent to those skilled in the art that modifications may be made in the embodiment chosen for illustration without departing from the spirit and scope of the invention.

We claim:

1. A method of automatically accessing web page information from internet addresses comprising the steps of:
 - a. automatically accessing web page information at a specified internet address;
 - b. downloading the web page information from the specified internet address; and
 - c. storing the web page information in order to allow a user to access the web page information while not connected to the specified internet address.
2. The method as claimed in claim 1 further comprising the step of forming a connection between a system capable of accessing a remote internet server and the remote internet server.
3. The method as claimed in claim 2 wherein the steps of automatically accessing and downloading are performed periodically at predetermined intervals.
4. The method as claimed in claim 3 wherein the system is a computer system.
5. The method as claimed in claim 4 wherein the computer system includes a memory device for storing the web page information.
6. The method as claimed in claim 5 further comprising the step of erasing a previous version of the web page information when a more recent version of the web page information is obtained.
7. A method of automatically accessing web page information from internet addresses comprising the steps of:
 - a. forming a connection between a system capable of accessing a remote internet server and the remote internet server;
 - b. automatically accessing web page information at a specified internet address; and

c. downloading the web page information from the specified internet address to the system.

8. The method as claimed in claim 7 further comprising the step of storing the web page information.

9. The method as claimed in claim 8 further comprising the step of providing the web page information to a user.

10. The method as claimed in claim 8 further comprising the step of erasing a previous version of the web page information when a more recent version of the web page information is obtained.

11. The method as claimed in claim 8 wherein the system is a computer system including a memory device for storing the web page information.

12. The method as claimed in claim 8 wherein the steps of forming a connection, automatically accessing and downloading are performed periodically at predetermined intervals.

13. An apparatus for automatically accessing web page information from internet addresses and providing the web page information to a user comprising:

a. a connection device configured for coupling to an internet server for forming a connection between the apparatus and the internet server; and

b. a controller coupled to the connection device for controlling the operation of the connection device to automatically form the connection between the apparatus and the internet server and download the web page information through the connection device from specified internet addresses.

14. The apparatus as claimed in claim 13 wherein the connection device is a modem.

15. The apparatus as claimed in claim 14 wherein the connection is formed over a telephone line.

16. The apparatus as claimed in claim 13 further comprising a memory device coupled to the connection for storing the web page information.

17. The apparatus as claimed in claim 16 further comprising a display device for displaying the web page information.

18. The apparatus as claimed in claim 17 further comprising means for programming coupled to the controller for entering the specified internet addresses.

19. The apparatus as claimed in claim 18 wherein the controller and the connection device automatically download the web page information periodically at predetermined intervals.

20. A computer system for automatically accessing web page information from internet addresses and providing the web page information to a user comprising:

a. a modem configured for coupling to a telephone line for forming a connection between the computer system and internet servers;

b. a controller coupled to the modem for controlling the modem to automatically form the connection between the computer system and the internet servers and download the web page information from specified internet addresses; and

c. a storage device coupled to the modem for storing the web page information.

21. The computer system as claimed in claim 20 further comprising a display device for displaying the web page information.

22. The computer system as claimed in claim 21 wherein the controller and the modem automatically download the web page information periodically at predetermined intervals.

23. The computer system as claimed in claim 22 wherein the storage device is a hard disk drive.

* * * * *

Von-Buhr, Maria

From: Goldberg, Gerald
Sent: Tuesday, June 23, 1998 7:52 AM
To: Lee, Thomas; Von-Buhr, Maria; Shaw, Gareth
Cc: Goodman, Talya
Subject: FW: Civil Action No. 98-CV-209 (Compton's)

Please see below. If you have anything, bring to my Office. If you know any names that should be supplied, let me know ASAP. Thanks

From: Godici, Nicholas
Sent: Monday, June 22, 1998 5:07 PM
To: Goldberg, Gerald
Subject: FW: Civil Action No. 98-CV-209 (Compton's)

Jerry, can you take the lead on responding for patents on this? Let me know if we need to meet.
Thanks

From: Buchanan, Karen
Sent: Monday, June 22, 1998 4:35 PM
To: Goldberg, Gerald; Barrett, Lee; Stoner, Bruce; Lee, Jameson; Kunin, Stephen; Godici, Nicholas; Kazenske, Edward; Goffney, Lawrence J.; Lehman, Bruce; Dickinson, Todd
Cc: Isacson, Linda; Kramer, Kevin
Subject: Civil Action No. 98-CV-209 (Compton's)

The above-identified case (a 35 U.S.C. § 145 action) relates to the PTO's reexamination (90/003,270) of the Compton's multimedia patent (5,241,671). We are currently conducting discovery in the case. In order to assist us in this effort, we request the following information:

1. All documents and/or files in your possession relating to either the reexamination or the initial examination of the patent.
2. The names of any individuals in your organization, not identified above, who were involved in either the reexamination or the initial examination of the patent.

If possible, we would like the names of individuals under 2. above as soon as possible (so that we may contact them).

Please provide the documents responsive to 1. above no later than COB Monday, July 6, 1998. They should be hand-delivered to the Solicitor's Office, Crystal Park 2, Suite 918.

If you do not possess any documents or files, please notify us of that fact via e-mail no later than COB Monday, July 6, 1998.

Thanks...kamb

cl. 2-22-92 / lpd. 2-30 / wdd. 4-10

Serial # 426,917

3rd action

cancels cl. 54, 63 & 74-80

w/D

amends cl. 6, 10, 13, 24, 26, 29, 34, 45, 54, 55, 58, 72 & 74

GIST same

AMDT (ENTERED)

REFERENCE(S): 4,945,476 (102/103) APS Text Search & Retrieval

cited: 4,939,689 Manual (103, combined)

cl. 11-12-92 / lpd. 11-21

4th action

retpd. 12-1 / wdd. 12-2

1st on final

- amends cl. 6, 72 & 73

GIST: same + info. re: independent storage - through entry path of independent storage - variations of related info.

(ALLEGEDLY INDEPENDENT)

REFERENCE(S): 4,914,586 (103)

cited: 5,159,669 5,065,345 4,998,248 4,982,344
4,931,950

cl. 4-15-93 / wdd.

5th action

independent!

- cancels 2, 3, 6 & 72

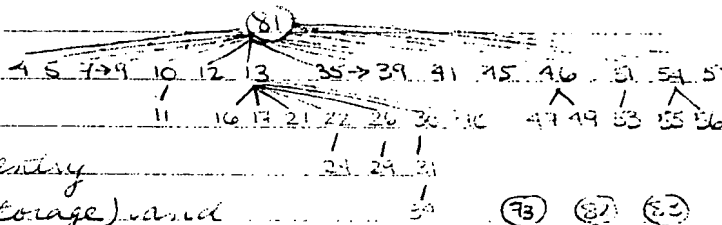
- adds cl. 81-83

- amends cl. 4, 5, 7-10, 12, 13, 16, 21, 22, 24, 26, 30, 34-41, 45, 46, 49, 51, 54, 57,

GIST: same +

in phases on

the plurality of entry
paths (one per storage) and
the accessibility of info. in
storage other than that associated with current entry



REFERENCE(S): none

cited: none

** ALLOWED - 4/15/93

(P.N. 5,241,671)

Serial #426, 917

Serial #426, 917

1st action

filing date: 10-26-89

auth: CK, signed & dated (joint)

foreign priority: none claimed

related application(s): none specified

title: OK "Multimedia Search System"

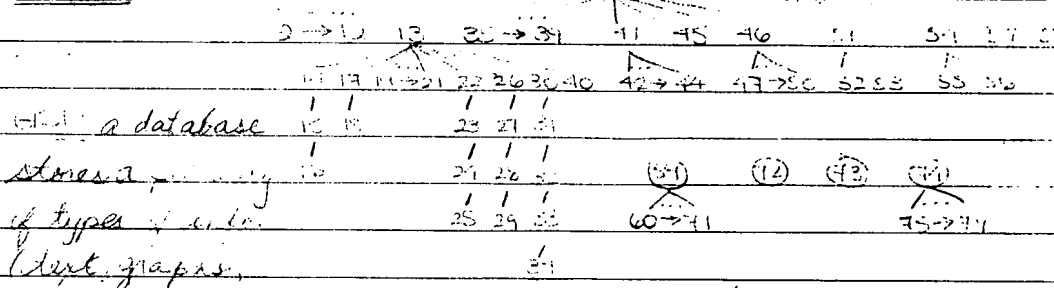
abstract: OK

drawings(s): informal, sly to by dman, no obvious errors

prior art: YES 1419 form (x 5), YES copies, NO cites in spec.

specification: no major problems

claims(s): 1-79



indices, etc., in individually accessible storage means, wherein indications of related information is stored & provided to a user, with an ability to access the same.

REFERENCES: 4,805,134 (102/103)

101 rejection

Cited: 4,923,314 4,916,655 4,829,423 4,811,217

4,422,158 4,831,610 4,829,169 4,814,972

4,774,050 4,774,596 4,758,955 4,757,302

4,729,043 4,717,971 4,685,001 4,587,635

EP 272,158 (6/88) EP 172,357 (2/86) 14 articles (good)

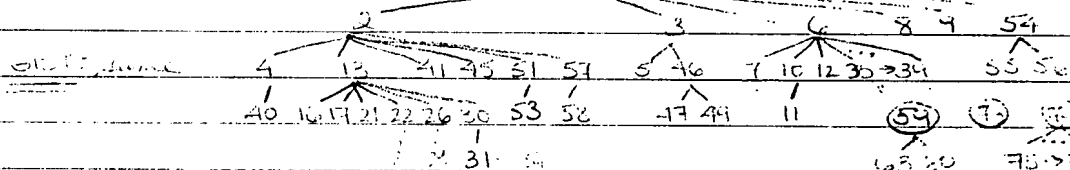
cl. 12-23, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79

2nd action
final!

- cancels cl. 1, 14, 15, 16-20, 23, 25, 27, 28, 32, 33, 42-44, 48, 50, 52, 60-62 & 64-71

- amends cl. 2-13, 16, 17, 21, 22, 24, 26, 30, 31, 34-41, 45-47, 49, 51, 54-59, 63 & 72-74

- adds cl. 80 (374, 29, 34)



REFERENCES: 4,805,134 (102, again)

Alleged prior art


[Excite Home](#) | [News Home](#)

[Click Here!](#)
[Visit Amazon.com's Office 2000 Store](#)
[News Home](#) [Top News](#) [World](#) [Business](#) [Sports](#) [Entertainment](#) [Tech](#) [Odd](#) [More](#)
[AP](#) • [Reuters](#) • [CBS.MW](#) • [Canada](#) • [Dow Jones](#) • [Biz Report](#) • [by Industry](#) • [Photos](#) • [My Stocks News](#)

Dickens-Soeder2000 Extends Deadline for Y2K Licensing Program; Prospective Licensees Have An Additional Month To Obtain License For \$50,000

Updated 7:30 AM ET May 1, 2000

IRVINE, Calif. (BUSINESS WIRE) - Dickens-Soeder2000 announced today that it has extended the deadline for its Y2K licensing program, effective now through May 31, 2000.

Prospective licensees have an additional month to obtain a license for a one-time lump sum payment of \$50,000. Other dates stated in the original licensing offer are also extended by one month, excepting the receipt of a U.S. Patent & Trademark Office (PTO) action.

The extension was granted in response to Fortune 500 and Information Week 500 companies seeking to review documents that had only recently been posted on the Dickens-Soeder2000 Web site (www.dickens-soeder2000.com) and that of their legal counsel, Levin & Hawes (www.levinhawes.com).

"We believe that our effort to conduct a licensing program electronically via the Internet is a first," said William C. Cray, a partner with the Laguna Beach, Calif. law firm of Levin & Hawes which represents Dickens-Soeder2000. "And, with many prospective licensees asking for more time to evaluate the materials just posted on our Web sites, we thought it only fair to extend our original deadline by a month."

Information that can currently be found on the two Web sites includes: the Dickens and Soeder issued patents and their file histories; the chain of title for the patents to Dickens-Soeder2000; the allowed claims in two pending Soeder patent applications, about to issue; the proposed license agreement; the response of Dickens-Soeder2000 to the Commissioner Ordered reexamination of the Dickens patent in the PTO; the anonymously filed Petition for Reexamination in the PTO; and the application to the PTO to reissue the Dickens patent.

Dickens-Soeder2000 has offices in Irvine and Columbia, Md. Questions about the company's licensing program should be directed to William C.

register
•com

[Click on our sponsors!](#)

AP Business News

[FBI Probes E-mails To Virus Creator](#)

[Asian Nations Back Currency Plan*](#)

[Boeing, Airbus in Race for Orders](#)

[Microsoft Settlement Predicted](#)

[Reprieve Offered on Privacy Plan](#)

[Cisco To Buy ArrowPoint for \\$6.1B](#)

[Conseco Sells \\$1.5B in Loans](#)

[Editors, Publishers To Meet in NYC](#)

May 6
[S.F. Newspaper Trial Taking Toll](#)

[Euro Becomes Early Embarrassment](#)

May 5
[Lawmakers Insulate Tobacco Cos.](#)

* Story has photo

[Printer-friendly format](#)
[Send this story to a friend](#)

Cray at 949/497-7676.

Contact: Communication Art Forms for Levin & Hawes Kathy Pinckert,
310/836-8355

Archive: [Sun May 7](#) [Sat 6](#) [Fri 5](#) [Thu 4](#) [Wed 3](#) [Tue 2](#) [Mon 1](#)

News Search

[New!](#) [Help](#) [Add URL](#) [Advertise on Excite](#) [Excite Affiliates](#) [Press Releases](#) [Jobs@Excite](#)
Copyright © 2000 At Home Corporation. All rights reserved. [Disclaimer](#) and [Privacy Statement](#)



How can I convince management that Linux will work in our enterprise setting?

O'REILLY NETWORK

OREILLY.COM

Dale Dougherty's Weblog

[HOME](#)
[Who Am I?](#)
[Discuss](#)
[Stories](#)

MEMBERS

[Join Now](#)
[Login](#)



Pei Wei to Marc Andreessen -- 02/03/93

Posted by Dale Dougherty, 3/30/00 at 4:43 PM.

A mail message from Pei Wei to Marc Andreessen updating him on new feature Viola, including the embedded object

Date: Mon, 8 Feb 93 17:23:34 -0800
 From: wei (Pei Y. Wei)
 Message-Id: <9302090123.AA18014@xcf.Berkeley.EDU>
 To: marca@ncsa.uiuc.edu
 Subject: stuff in new violaWWW
 Cc: wei@xcf.Berkeley.EDU
 Status: O

> So what does the new Viola have in it?? Huh huh? Gotta keep
 > the competition... :-)
 >
 > Marc

Hello, Marc.

Well, the new violaWWW will support SGML documents. It does one particular now (mine:-), but it's designed to be extensible to other DTDs. So, when the W DTD is stabilized, I'll add it... Unlike the old HTML widget, the new "widget" can embed bitmaps and viola objects into the document page...

The old HTML widget hasn't changed much, so old HTML will continue to work.

I'm looking to make a GIF widget (or something more compact than XBM/XPM got any suggestions?). I'd eventually like to add graphic widgets such as gplot and but this is low priority for us (ORA), and I imagine a higher priority for you (NCSA). We're concentrating on text for now.

A significant new capability is to be able to embed any viola objects inside a document. Indeed, an viola application can be considered as a WWW "document". This means that a document can have much GUIs in it (ie: entry forms, interactive tutorial scripts, etc). Until some common SGML-based GUI DTD (ick!) comes out, using this capability can introduce quite a bit of violaism. But this is no worse than using LaTeX with the WWW. And it'd make it possible for documents to be very programmatic and interactive (yes, there's a security issue, but I'm going to assume people are nice).

RTW if you're interested, it's a possibility that I will make viola into a library and

DALE, if you're interested, it's a possibility that I will make Xerox into a widget, as a widget to stick into XMosaic. I noticed that my boss, Dale Dougherty, had wa no time in passing this idea to you some time ago. We're trying to cooperate wit others as much as possible and practically.

 Reply ►



Who runs Linux on Mac hardware?

O'REILLY NETWORK

OREILLY.COM

Dale Dougherty's Weblog

[HOME](#)
[Who Am I?](#)
[Discuss](#)
[Stories](#)

MEMBERS

[Join Now](#)
[Login](#)



oreillynet.com

Viola Is a Repository of Prior Art for The Web

Posted by [Dale Dougherty](#), 3/30/00 at 4:50 PM.

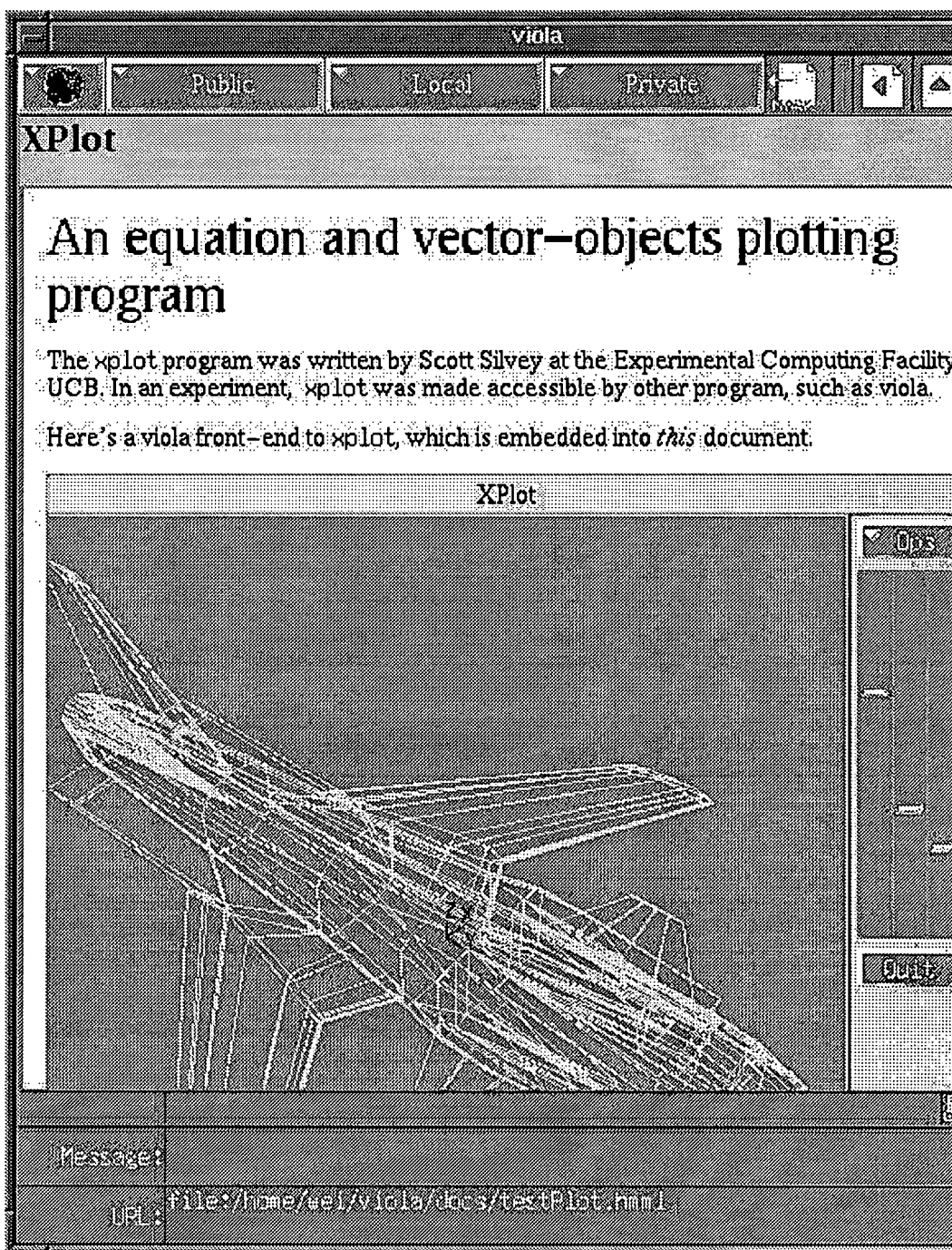
I had a visit today from Pei Wei, who was the developer of Viola, the first graphic browser. Viola was an interpreted language and graphical toolkit for creating hype applications, and Pei used Viola in 1992 to build a Web browser.

Pei and I began working together at O'Reilly in 1991. Much of the work on Viola was experimental in nature; when I met him, Pei was hosting his work at the Experimental Computing Facility (XCF) at University of California at Berkeley. I was trying to find online books at the time, and I hired Pei so that we could research the new possibilities were emerging on the Internet. One day, Pei showed up and said he had implemented a browser; he had read the HTTP specs and the HTML documents from Tim Berners-Lee's project. (Tim's www browser was line-oriented.) What's more he extended it to include text and bitmapped graphics. Then, it was like seeing HyperCard suddenly reaching the Internet for information. Pei's contributions helped Tim Berners-Lee realize his vision of the Web, although his work came to be eclipsed by the Mosaic team.

Strangely enough, one of the reasons we reconnected recently concerned patents. Patent #5838906 has to do with controlling embedded hypermedia applications inside a browser. A person named Michael Doyle applied for this patent in 1994, and he got the patent. Now he is suing Microsoft (going after the big fish first) saying that ActiveX controls and other things infringe on his patent. The lawsuit set Microsoft off on a search for products they uncovered Viola.

In May 1993, as part of the ongoing experimentation with Viola, Pei had created a demo that showed how he could control another application inside his browser. He used an application embedded in a web page, which communicated with a separate program called xplot. The application displayed a wireframe model of a jet aircraft on the web page. The controls in the application allowed you to change the perspective of the view and rotate the aircraft; the application sent messages back to the program (not manipulating the display directly.)

According to Pei, he had put this demo together (with an associate, Scott Silvey) before he was bringing some folks over from Sun Microsystems to see Viola. Pei and Scott were busy at night making the Xplot demo work. (See the screenshot below.)



Interestingly, the folks from Sun that were visiting were from the Oak project, which was in the Nomadic group, and later became Java. (There were many aspects of Viola similar to Java -- for whatever that's worth.)

After the meeting, we even gave a copy of the Viola source code to the Sun representative, which they took back to their labs to compile. This might seem strange to some, and I know that others have asked me about it. It was just the way Pei worked, which was to work with others.

It is this meeting that provides substantial evidence that we had created the ability to embed an application from within a web browser and had demonstrated it to others.

full year before Doyle filed his patent claim. (The year time-period is important for patent.)

We have also had to go through old email archives to find support for the fact that known to others and to create a timeline for when various features were introduced. A very interesting email from Pei Wei to Marc Andreessen -- 02/03/93.

Pei and I agreed to provide information to Microsoft to help fight this software patent. It is ironic that while Microsoft fights one patent, they also hold a patent on stylesheets for the Web. Viola, which used stylesheets in its browser, could be used to fight the Microsoft patent. This just points out how silly the whole business of patents really is.

In talking about this with Pei, I thought that Viola's greatest value may yet to be realized. It represents a valuable repository of prior art with which to fight all kinds of Web-related patents. Pei has set up the Viola archive, which contains the source code and documentation and he and I are going to improve the archive. We want to provide whatever information we can to help others dispute web-related patents. Again, because Viola's browser development preceded even Mosaic, any feature we find in it can be said to be public knowledge and used to dispute any claim that any later arrival to the Web party invented it.

